

(11)特許出願公開番号

特開平8-63442

(43)公開日 平成8年(1996)3月8日

(51) Int.Cl.⁶

識別記号

庁内整理番号

FI

技術表示箇所

G O 6 F 15/163

G 0 6 F 15/ 16

3 1 0 v

審査請求 未請求 請求項の数4 OL (全 49 頁)

(21)出願番号 特願平6-202071

(22)出願日 平成6年(1994)8月26日

(71) 出願人 000004226

日本電信電話株式会社

東京都新宿区西新宿三丁目19番2号

(72)発明者 山田 茂樹

東京都千代田区内幸町1丁目1番6号 日

本電信電話株式会社内

(72)発明者 丸山 勝己

東京都千代田区内幸町1丁目1番6号 日

本電信電話株式会社内

(72)発明者 久保田 稔

東京都千代田区内幸町1丁目1番6号 日

本電信電話株式会社内

(74) 代理人 弁理士 磯村 雅俊 (外1名)

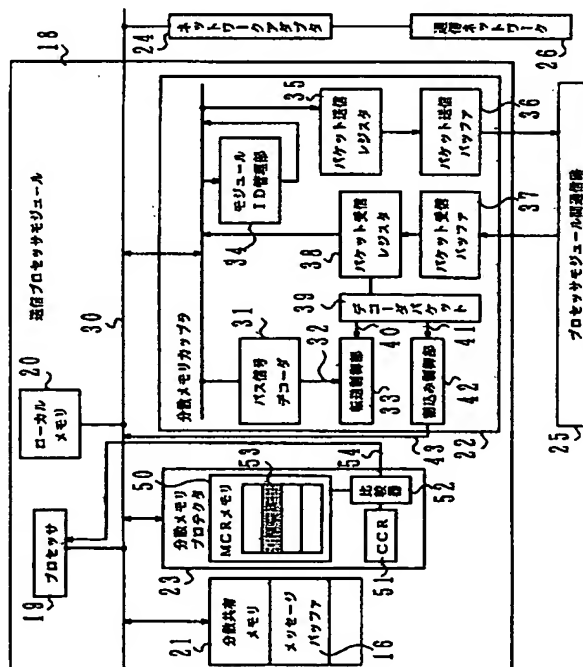
最終頁に続く

(54) 【発明の名称】 マルチプロセッサシステム

(57) 【要約】

【目的】 カーネル等のソフトウェアのオーバヘッドを削減し、かつ転送遅延時間が短く、処理効率の高いマルチプロセッサシステムのメッセージ転送を実現する。

【構成】 各プロセッサモジュールは分散共有メモリと分散メモリカップラとを備え、分散メモリカップラは、送信側の分散共有メモリに書き込みがあった時点で、アドレスを共有する他のプロセッサモジュールに書き込み情報を転送し、受信側のメモリにコピーする。MB管理マップを分散共有メモリに配置することにより、送信側分散共有メモリに書き込み、受信プロセッサモジュールが同一アドレスロケーションから読み出すだけで通信の同期をとることができる。返事は、MBディスクリプタとMB管理マップに書き込まれると、送信側が読み出すことにより、状態を送信側に伝達できる。また、受信側分散共有メモリをF I F O構成にすることにより、処理要求の検出を簡単に行うことができる。



1

【特許請求の範囲】

【請求項 1】複数のプロセッサモジュールから構成されたマルチプロセッサシステムにおいて、

上記プロセッサモジュールには、

各プロセッサモジュール間で共通のアドレスを有し、送信プロセッサモジュールと受信プロセッサモジュールの組み合わせで指定される管理単位毎に分割された分散共有メモリと、

上記送信プロセッサモジュールの分散共有メモリ（以下、送信側分散共有メモリ）とアドレスを共有する受信プロセッサモジュールの識別情報を記憶し、該送信側分散共有メモリへの書き込みが発生すると、上記受信プロセッサモジュールの識別情報で指定された受信プロセッサモジュールに、書き込みアドレスと書き込みデータを送信するとともに、受信した書き込みアドレスと書き込みデータをもとに受信プロセッサモジュールの分散共有メモリ（以下、受信側分散共有メモリ）の、送信側と同一のアドレスロケーションにコピーを行う分散共有メモリ制御手段と、

送信するメッセージの宛先情報をもとに受信プロセッサモジュールを特定し、上記分散共有メモリ上の送信プロセッサモジュールと受信プロセッサモジュールの組み合わせで指定されるメッセージバッファを捕捉するメッセージバッファ管理手段とを有し、

送信プロセッサモジュールから受信プロセッサモジュールにメッセージを送信する場合に、送信プロセッサモジュールは、上記メッセージバッファ管理手段により上記受信プロセッサモジュール対応の送信メッセージバッファを捕捉し、該送信メッセージバッファに上記メッセージを書き込み、

該送信プロセッサモジュールと受信プロセッサモジュールとが同一モジュールである場合には、上記受信プロセッサモジュールは、上記送信メッセージバッファから、直接、メッセージを読み出すことにより、同一プロセッサ内メッセージ通信を行い、

送信プロセッサモジュールと受信プロセッサモジュールが異なるモジュールである場合には、送信側の分散共有メモリ制御手段が、上記送信メッセージバッファのアドレスをもとに受信プロセッサモジュール識別情報を取り出し、該識別情報の受信プロセッサモジュールに送信メッセージバッファアドレスと書き込みデータを転送し、受信側の分散共有メモリ制御手段は、受信した上記送信メッセージバッファのアドレスと上記書き込みデータをもとに受信側の分散共有メモリの、送信側と同一アドレスの受信メッセージバッファにコピーを行い、上記受信プロセッサモジュールでは、上記受信メッセージバッファからメッセージを読み出すことにより、異なるプロセッサ間のメッセージ通信を行うことを特徴とするマルチプロセッサシステム。

【請求項 2】請求項 1 に記載のマルチプロセッサシステ

2

ムにおいて、上記受信側の分散共有メモリ制御手段は、さらにアドレスを共有する送信プロセッサモジュールの識別情報を記憶し、

上記送信プロセッサモジュールでは、該送信プロセッサモジュールと受信プロセッサモジュールの組み合わせで指定される分散共有メモリ上の送信制御エリアに、メッセージ制御情報あるいはメッセージバッファ管理情報を含む制御データを書き込み、

送信プロセッサモジュールと受信プロセッサモジュールとが同一モジュールである場合には、該受信プロセッサモジュールは、上記送信制御エリアから、直接、制御データを読み出してメッセージあるいはメッセージバッファに関する制御内容を認識し、対応する処理を実行した後、上記送信制御エリアに返答制御データを書き込み、上記送信プロセッサモジュールは、上記送信制御エリアから上記返答制御データを読み出して受信側での処理結果を認識することにより、同一プロセッサモジュール内双方向通信を行い、

送信プロセッサモジュールと受信プロセッサモジュールが異なるモジュールである場合には、上記送信側の分散共有メモリ制御手段は、上記送信制御エリアのアドレスをもとに受信プロセッサモジュール識別情報を取り出し、該送信制御エリアアドレスと書き込みデータを上記受信プロセッサモジュールに転送し、

上記受信側の分散共有メモリ制御手段は、受信した上記送信制御エリアアドレスと上記書き込みデータをもとに受信側分散共有メモリの、送信側と同一アドレスの受信制御エリアにコピーを行い、

上記受信プロセッサモジュールは、上記受信制御エリアから制御データを読み出して、メッセージあるいはメッセージバッファに関する制御内容を認識し、対応する処理を実行した後、上記受信制御エリアに返答制御データを書き込むと、

受信側の分散共有メモリ制御手段は、上記受信制御エリアアドレスをもとに、上記送信プロセッサモジュール識別情報を取り出し、該識別情報をもとに上記受信制御エリアアドレスと上記返答制御データを上記送信プロセッサモジュールに転送し、

上記送信側の分散共有メモリ制御手段は、受信した上記受信制御エリアアドレスと上記返答制御データをもとに送信側分散共有メモリの、受信側と同一アドレスの送信制御エリアにコピーを行い、

上記送信プロセッサモジュールでは、上記送信制御エリアから上記返答制御データを読み出して、受信側での処理結果を認識することにより、プロセッサモジュール間双方向通信を行うことを特徴とするマルチプロセッサシステム。

【請求項 3】複数のプロセッサモジュールから構成されたマルチプロセッサシステムにおいて、

上記プロセッサモジュールには、

50

3

各プロセッサモジュール間で共通の同一アドレスを有し、受信プロセッサモジュール対応の管理単位毎に分割された分散共有メモリと、

送信側分散共有メモリとアドレスを共有する受信プロセッサモジュールの識別情報を記憶し、送信側分散共有メモリへの書き込みが発生すると、上記受信プロセッサモジュール識別情報で指定された受信プロセッサモジュールに書き込みアドレスと書き込みデータを送信するとともに、受信した書き込みアドレスと書き込みデータをもとに受信側分散共有メモリの、送信側と同一のアドレス

ロケーションに書き込みを行う分散共有メモリ制御手段とを有し、
上記受信側の分散共有メモリのエリアはFIFOメモリで構成され、

複数の送信プロセッサモジュールから1つの受信プロセッサモジュールに処理要求を通知する際に、上記送信プロセッサモジュールと受信プロセッサモジュールが同一のモジュールである場合には、上記送信プロセッサモジュールでは、送信側分散共有メモリ上の自プロセッサモジュール宛処理要求FIFOエリアに、処理要求データ

を書き込み、
上記送信プロセッサモジュールと受信プロセッサモジュールが異なるモジュールである場合には、上記送信プロセッサモジュールでは、上記送信側分散共有メモリ上の受信プロセッサモジュール宛処理要求FIFOエリアに、処理要求データを書き込み、

送信側分散共有メモリ制御手段は、上記処理要求エリアのアドレスをもとに受信プロセッサモジュール識別情報を取り出し、上記処理要求エリアアドレスと書き込みデータを受信プロセッサモジュールに転送し、

受信側分散共有メモリ制御手段は、受信した上記処理要求FIFOエリアアドレスと上記書き込みデータをもとに受信側分散共有メモリの、送信側と同一アドレスの上記処理要求FIFOエリアに書き込むことにより、上記処理要求FIFOエリアに複数の処理要求データを到着順に蓄積し、

上記受信側プロセッサモジュールでは、受信側分散共有メモリ上の上記要求FIFOエリアから複数の上記処理要求データを順次読み出すことにより、処理要求を検出することを特徴とするマルチプロセッサシステム。

【請求項4】請求項1ないし3のいずれか1つに記載のマルチプロセッサシステムにおいて、上記送信側分散共有メモリ制御手段は、送信側分散共有メモリの連続する複数のアドレスロケーションに複数のデータが連続的に書き込まれた場合、上記複数のデータの先頭アドレスと複数のデータとをまとめて受信プロセッサモジュールに一括送信し、

受信側分散共有メモリ制御手段は、一括受信した上記先頭アドレスと上記複数のデータをもとに受信側分散共有メモリ上の同一アドレスロケーションに連続的にコピー

4

を行うことを特徴とするマルチプロセッサシステム。

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は、複数のプロセッサモジュールから構成されるマルチプロセッサシステムに関し、特に転送遅延時間が短く、プロセッサモジュール間のリソース競合を回避でき、処理効率の高いメッセージ転送が可能なマルチプロセッサシステムに関するものである。

【0002】

【従来の技術】複数のプロセッサを組み合わせるマルチプロセッサシステムを用いて、複数のオブジェクト（並列実行の単位となるプロセス）の間でメッセージを通信しながら処理を進めるオブジェクト指向分散処理方式が提案されている。例えば、T. Shimizu et al. "Low-Latency Communication Support for the AP 1000", Proceedings of the 19th International Symposium on Computer Architecture, pp. 288~297, 1992には、上記のようなオブジェクト指向分散処理方式が記述されている。図2および図3は、上記文献による従来例を示すシステム構成図およびプロセッサ間メッセージ通信方法を示すシーケンスチャートである。図2において、1-1, 1-2はマルチプロセッサシステムにおけるプロセッサモジュール（PM）、2-1, 2-2はプロセッサモジュール内のプロセッサ、3-1, 3-2はそれぞれ対応するプロセッサ1-1, 1-2から読み書きアクセスが可能なローカルメモリ、4-1, 4-2はPM間でメッセージ転送を行うためのDMA（Direct Memory Access）コントローラ、5はプロセッサ間通信路、6はローカルメモリ3-1上に設けられたメッセージバッファエリア、7は同じくローカルメモリ3-2上に設けられたメッセージバッファエリアである。

【0003】図2に示すマルチプロセッサシステムにおいて、PM1-1からPM1-2にメッセージを転送する場合のシーケンスを、図3により説明する。S1はメッセージバッファ要求、S2はメッセージバッファエリア6の確保、S3はメッセージバッファエリア6へメッセージを書込み、S4は送信要求、S5はDMAコントローラ4-1に制御情報設定、S6はDMA起動、S7はメッセージバッファエリア6からプロセッサ間通信路5へ転送、S8はプロセッサ間転送、S9は受信一時バッファに記憶、S10は割込み、S11はメッセージバッファエリア7の確保、S12はDMAコントローラ4-2に制御情報設定、S13はDMA起動、S14は受信一時バッファからメッセージバッファエリア7に転送、S15は割込み、S16は受信先の特典、S17は受信オブジェクト起動、S18はメッセージの読出しの各ステップを示している。図3におけるプロセッサモジュール1-1の送信オブジェクト10からメッセージバッファ要求S1があると、オペレーティングシステムの

5

中心的プログラムであるカーネル 11-1 がステップ S 2 でメッセージバッファエリア 6 をローカルメモリ 3-1 内に確保し、送信オブジェクト 10 に通知する。送信オブジェクト 10 は、ステップ S 3 でメッセージバッファエリア 6 にメッセージを書き込む。図 2 においては、太線矢印 L 10 で示されている。送信オブジェクト 10 はカーネル 11-1 にステップ S 4 で送信要求を出し、カーネル 11-1 はステップ S 5 でメッセージバッファエリア 6 の先頭アドレス (図 2 の ADR 1) とメッセージサイズ (図 2 の n)、および転送先のプロセッサモジュール番号等の制御情報を DMA コントローラ 4-1 に設定する。これは、図 2 においては、太線矢印 L 11 で示される。カーネル 11-1 は、ステップ S 6 で DMA コントローラ 4-1 を起動し、DMA コントローラ 4-1 は設定された制御情報に従って、ステップ S 7、S 8 でメッセージをメッセージバッファエリア 6 から順次読み出し、プロセッサ間通信路 5 を経由して受信側 PM 1-2 の DMA コントローラ 4-2 に転送する。これは、図 2 では、太線矢印 L 12 で示される。

【0004】DMA コントローラ 4-2 は、ステップ S 9 でメッセージを一時記憶に蓄積すると同時に、並行してステップ S 10 でプロセッサ 2-2 のカーネル 11-2 に割り込みをかける。これは、図 2 においては、太線矢印 L 13 で示される。カーネル 11-2 は、ステップ S 11 で受信側 PM 1-2 のローカルメモリ 3-2 上にメッセージバッファエリア 7 のアドレス (図 2 の ADR 2) とメッセージサイズ (図 2 の n) 等の制御情報を設定する。カーネル 11-2 がステップ S 13 で DMA 起動を行うと、DMA コントローラ 4-2 はステップ S 14 でメッセージをメッセージバッファエリア 7 に転送し (図 2 の太線矢印 L 14)、ステップ S 15 でカーネル 11-2 に転送終了を通知する。カーネル 11-2 は、ステップ S 16 でメッセージ内に書込まれた宛先を見て、図 2 の受信オブジェクト 12 を特定し、ステップ S 17 で起動する。受信オブジェクト 12 がステップ S 18 でローカルメモリ 3-2 内のメッセージバッファ 7 からメッセージを読み出す (これは図 2 の太線矢印 L 15)。以上の手順により、PM 間のメッセージ転送が行われる。

【0005】

【発明が解決しようとする課題】図 3 においては、送信オブジェクト 10 がステップ S 3 でメッセージバッファエリア 6 に書き込んだ段階で、受信 PM 宛のメッセージ転送をできるだけ早く開始することが望ましい。また、受信側においても、図 3 のステップ S 9 でメッセージを受信した段階で、できるだけ早く受信オブジェクト 12 にメッセージを引き渡すことが望ましい。しかしながら、従来の技術では、図 3 に示すように、ステップ S 5 とステップ S 12 の DMA 起動準備処理や、ステップ S 11 における受信側でのメッセージバッファエリアの捕

6

捉、ステップ S 10、S 15 における割り込み処理等があり、メッセージ送受信を終了させるまでには多くのカーネル処理が必要である。その結果、メッセージ転送遅延の増加とカーネルの処理オーバーヘッド増加を招くという問題がある。特に、プロセッサ間通信路 5 の転送速度が大きい場合には、上記処理オーバーヘッドによる転送遅延の比率は相対的に増加するため、処理オーバーヘッドの削減と転送遅延時間の短縮が必須の条件となってくる。また、ステップ S 10、S 15 では、送信 PM からのメッセージ到着を通知する方法として割り込みを用いている。しかし、割り込み要求を受け付けるためには、現在実行中の情報を退避して、割り込み処理に切り替えるためのカーネルの処理オーバーヘッドが一般に大である。従って、割り込み方式を多数の PM を含む超並列システムに適用した場合には、単位時間当りの割り込み回数の増加により、処理オーバーヘッドが非常に大となって、転送遅延も増大するという問題が生じる。

【0006】割り込みを回避する方法としては、従来より、ポーリング方式と呼ばれる要求受け付け方式が用いられている。ポーリング方式は、プロセッサ側が外部からの処理要求を予め決められたメモリやレジスタ等に記憶させておき、処理が可能になると、プロセッサが処理要求を読み出してきて処理を開始する方法である。このように、ポーリング方式は、割り込み方式に比較して処理切り替えのオーバーヘッドが少ないが、処理要求を記憶する全てのメモリやレジスタを読み出す必要があるため、処理要求の発生頻度が少ない場合には処理要求の読み出しが空振りに終る、つまり処理要求が発生していない現象となることも多い。特に、多数の PM を有する超並列システムでは、処理要求の数および種類が多いため、それらの全てを定期的に読み出すための処理オーバーヘッドが大きく、空振りの回数も増加して処理効率が低下するという問題があった。また、本出願人は本願に先立って、【分散共有メモリシステム】(特願平 6-74669 号明細書および図面参照)を提案した。上記発明においては、分散共有メモリを送信プロセッサモジュールと受信プロセッサモジュールの組み合わせ毎に対応して分割し、各プロセッサモジュールからメッセージを送信する場合、自プロセッサモジュールを送信元とする受信プロセッサモジュール対応のエリアを使用する。その結果、異なる 2 つの送信プロセッサモジュールから同一受信プロセッサモジュールにメッセージ通信要求が同時に発生しても、互いに使用するメッセージバッファが重複することがなく、メッセージバッファ確保のための競合処理に伴う性能低下や性能ボトルネックを回避することができる。本発明の目的は、上記のような従来の課題を解決し、先願の発明を利用して、カーネル等のシステムソフトウェアのオーバーヘッドを削減するとともに、転送遅延時間を短縮して、処理効率の高いメッセージ転送を実現できるマルチプロセッサシステムを提供すること

にある。

【0007】

【課題を解決するための手段】上記目的を達成するため、本発明のマルチプロセッサシステムは、

(イ) 複数のプロセッサモジュール (18) から構成されたマルチプロセッサシステム (17) において、上記プロセッサモジュール (18) には、各プロセッサモジュール間で共通のアドレスを有し、送信プロセッサモジュールと受信プロセッサモジュールの組み合わせで指定される管理単位毎に分割された分散共有メモリ (21) と、上記送信プロセッサモジュールの分散共有メモリ

(以下、送信側分散共有メモリ 21) とアドレスを共有する受信プロセッサモジュールの識別情報を記憶し、該送信側分散共有メモリ (21) への書き込みが発生すると、上記受信プロセッサモジュールの識別情報で指定された受信プロセッサモジュールに、書き込みアドレスと書き込みデータを送信するとともに、受信した書き込みアドレスと書き込みデータをもとに受信プロセッサモジュールの分散共有メモリ (以下、受信側分散共有メモリ 21) の、送信側と同一のアドレスロケーションにコピーを行う分散共有メモリ制御手段 (22) と、送信するメッセージの宛先情報をもとに受信プロセッサモジュールを特定し、上記分散共有メモリ上の送信プロセッサモジュールと受信プロセッサモジュールの組み合わせで指定されるメッセージバッファ (16) を捕捉するメッセージバッファ管理手段 (200) とを有し、送信プロセッサモジュール (18) から受信プロセッサモジュール (18) にメッセージを送信する場合に、送信プロセッサモジュールは、上記メッセージバッファ管理手段 (200) により上記受信プロセッサモジュール対応の送信メッセージバッファ (180) を捕捉し、該送信メッセージバッファ (180) に上記メッセージを書き込み、該送信プロセッサモジュール (18) と受信プロセッサモジュール (18) とが同一モジュールである場合には、上記受信プロセッサモジュール (18) は、上記送信メッセージバッファから、直接、メッセージを読み出すことにより、同一プロセッサ内メッセージ通信を行い、送信プロセッサモジュールと受信プロセッサモジュールが異なるモジュールである場合には、送信側の分散共有メモリ制御手段 (22) が、上記送信メッセージバッファのアドレスをもとに受信プロセッサモジュール識別情報を取り出し、該識別情報の受信プロセッサモジュールに送信メッセージバッファアドレスと書き込みデータを転送し、受信側の分散共有メモリ制御手段は、受信した上記送信メッセージバッファのアドレスと上記書き込みデータをもとに受信側の分散共有メモリの、送信側と同一アドレスの受信メッセージバッファにコピーを行い、上記受信プロセッサモジュールでは、上記受信メッセージバッファからメッセージを読み出すことにより、異なるプロセッサ間のメッセージ通信を行うことを特徴

としている。

【0008】 (ロ) 上記受信側の分散共有メモリ制御手段は、さらにアドレスを共有する送信プロセッサモジュールの識別情報を記憶し、上記送信プロセッサモジュールでは、該送信プロセッサモジュールと受信プロセッサモジュールの組み合わせで指定される分散共有メモリ上の送信制御エリアに、メッセージ制御情報あるいはメッセージバッファ管理情報を含む制御データを書き込み、送信プロセッサモジュールと受信プロセッサモジュールとが同一モジュールである場合には、該受信プロセッサモジュールは、上記送信制御エリアから、直接、制御データを読み出してメッセージあるいはメッセージバッファに関する制御内容を認識し、対応する処理を実行した後、上記送信制御エリアに返答制御データを書き込み、上記送信プロセッサモジュールは、上記送信制御エリアから上記返答制御データを読み出して受信側での処理結果を認識することにより、同一プロセッサモジュール内双方向通信を行い、送信プロセッサモジュールと受信プロセッサモジュールが異なるモジュールである場合には、上記送信側の分散共有メモリ制御手段は、上記送信制御エリアのアドレスをもとに受信プロセッサモジュール識別情報を取り出し、該送信制御エリアアドレスと書き込みデータを上記受信プロセッサモジュールに転送し、上記受信側の分散共有メモリ制御手段は、受信した上記送信制御エリアアドレスと上記書き込みデータをもとに受信側分散共有メモリの、送信側と同一アドレスの受信制御エリアにコピーを行い、上記受信プロセッサモジュールは、上記受信制御エリアから制御データを読み出して、メッセージあるいはメッセージバッファに関する制御内容を認識し、対応する処理を実行した後、上記受信制御エリアに返答制御データを書き込むと、受信側の分散共有メモリ制御手段は、上記受信制御エリアアドレスをもとに、上記送信プロセッサモジュール識別情報を取り出し、該識別情報をもとに上記受信制御エリアアドレスと上記返答制御データを上記送信プロセッサモジュールに転送し、上記送信側の分散共有メモリ制御手段は、受信した上記受信制御エリアアドレスと上記返答制御データをもとに送信側分散共有メモリの、受信側と同一アドレスの送信制御エリアにコピーを行い、上記送信プロセッサモジュールでは、上記送信制御エリアから上記返答制御データを読み出して、受信側での処理結果を認識することにより、プロセッサモジュール間双方向通信を行うことも特徴としている。

【0009】 (ハ) 複数のプロセッサモジュールから構成されたマルチプロセッサシステムにおいて、上記プロセッサモジュールには、各プロセッサモジュール間で共通の同一アドレスを有し、受信プロセッサモジュール対応の管理単位毎に分割された分散共有メモリと、送信側分散共有メモリとアドレスを共有する受信プロセッサモジュールの識別情報を記憶し、送信側分散共有メモリへ

の書き込みが発生すると、上記受信プロセッサモジュール識別情報で指定された受信プロセッサモジュールに書き込みアドレスと書き込みデータを送信するとともに、受信した書き込みアドレスと書き込みデータをもとに受信側分散共有メモリの、送信側と同一のアドレスロケーションに書き込みを行う分散共有メモリ制御手段とを有し、上記受信側の分散共有メモリのエリアはFIFOメモリで構成され、複数の送信プロセッサモジュールから1つの受信プロセッサモジュールに処理要求を通知する際に、上記送信プロセッサモジュールと受信プロセッサモジュールが同一のモジュールである場合には、上記送信プロセッサモジュールでは、送信側分散共有メモリ上の自プロセッサモジュール宛処理要求FIFOエリアに、処理要求データを書き込み、上記送信プロセッサモジュールと受信プロセッサモジュールが異なるモジュールである場合には、上記送信プロセッサモジュールでは、上記送信側分散共有メモリ上の受信プロセッサモジュール宛処理要求FIFOエリアに、処理要求データを書き込み、送信側分散共有メモリ制御手段は、上記処理要求エリアのアドレスをもとに受信プロセッサモジュール識別情報を取り出し、上記処理要求エリアアドレスと書き込みデータを受信プロセッサモジュールに転送し、受信側分散共有メモリ制御手段は、受信した上記処理要求FIFOエリアアドレスと上記書き込みデータをもとに受信側分散共有メモリの、送信側と同一アドレスの上記処理要求FIFOエリアに書き込むことにより、上記処理要求FIFOエリアに複数の処理要求データを到着順に蓄積し、上記受信側プロセッサモジュールでは、受信側分散共有メモリ上の上記要求FIFOエリアから複数の上記処理要求データを順次読み出すことにより、処理要求を検出することも特徴としている。

【0010】(二)上記送信側分散共有メモリ制御手段は、送信側分散共有メモリの連続する複数のアドレスロケーションに複数のデータが連続的に書き込まれた場合、上記複数のデータの先頭アドレスと複数のデータとをまとめて受信プロセッサモジュールに一括送信し、受信側分散共有メモリ制御手段は、一括受信した上記先頭アドレスと上記複数のデータをもとに受信側分散共有メモリ上の同一アドレスロケーションに連続的にコピーを行うことも特徴としている。

【0011】

【作用】本発明においては、①送信プロセッサモジュールの分散共有メモリ制御手段に、分散共有メモリアドレスを共有する受信プロセッサモジュール識別番号を記憶することにより、送信側共有メモリへの書き込みが行われると、分散共有メモリ制御手段からそのアドレスに対応する受信プロセッサモジュール識別番号を読み出し、対応する受信プロセッサモジュールに書き込み情報を送出すると、受信側の分散共有メモリ制御手段は、受信側分散共有メモリの同一アドレスロケーションにコピーを

行う。送信プロセッサモジュール内の分散共有メモリは、受信プロセッサ対応にメッセージバッファ群が分割されており、ある受信プロセッサ対応のメッセージバッファにメッセージが書き込まれると、分散共有メモリ制御手段により、対応する送受信プロセッサモジュール内の分散共有メモリの同一ロケーションのメッセージバッファにコピーが行われる。このように、プロセッサモジュール間で高速にオーバーヘッドの少ないメッセージ通信が実現できる。

②また、受信側の分散共有メモリ制御手段にも、分散共有メモリアドレスを共有する送信プロセッサモジュール識別番号を記憶しておき、これにより受信側分散共有メモリのメッセージ制御エリアにメッセージ制御情報を書き込むと、送信側分散共有メモリにコピーされることにより、メッセージ制御に必要な双方向通信が行える。

【0012】③さらに、受信側分散共有メモリの処理要求エリアをFIFOメモリにすることにより、送信側分散共有メモリに書き込んだ処理要求が、受信側分散共有メモリのFIFOに順次蓄積される。従って、複数の送信側プロセッサモジュールが同時に各分散共有メモリの同一FIFOロケーションに処理要求を書き込んだ場合でも、受信側の分散共有メモリのFIFOで自動的に順序付けが行われるので、全ての処理要求がFIFOに順次書き込まれる。受信側は、分散共有メモリ内の要求登録FIFOを順次読み出すだけで、発生した要求のみを効率よく取り出せる。従って、従来のポーリング処理や割り込み処理で必要とされた処理オーバーヘッドを大幅に削減することができる。

④また、先願である特願平6-74669号明細書および図面に記載のものに比較すると、分散共有メモリが全てのプロセッサモジュール間で共通のアドレスが付加され、送信プロセッサモジュールと受信プロセッサモジュールの組み合わせで指定される管理単位毎に分割されている点、および分散共有メモリを、自プロセッサモジュール内のローカル転送データのための送受信用と、異なるプロセッサモジュール間のリモート転送データのための送受信用のエリアに分割している点では共通している。しかし、上記明細書では、送信プロセッサモジュールが送信および受信メッセージバッファを捕捉し、受信プロセッサモジュールがそれらを解放し、受信プロセッサモジュールで再利用するか、あるいは送信プロセッサモジュールに再利用を許可する方法を用いている。その結果、2つのプロセッサモジュール間で双方向のメッセージ通信のトラヒックがアンバランスな場合、メッセージバッファの過不足が生じ易く、その制御手順も複雑で処理オーバーヘッドも大きい。これに対して、本発明では、送信および受信メッセージバッファの捕捉、解放は送信プロセッサモジュールで一元的に行われ、送信側プロセッサモジュールで常に再利用されるため、制御手順も簡単である。また、上記明細書では、送信メッセージ

11

バッファから受信メッセージバッファへのコピーは、送信メッセージバッファにメッセージが書かれた後、メッセージ送信要求が発せられた時点で開始される。これに対して、本発明では、送信メッセージバッファから受信メッセージバッファへのコピーは、送信メッセージバッファにメッセージが書き込まれた瞬間から開始されるため、メッセージ転送遅延時間を短縮することができる。そして、本発明では、さらに受信側の分散共有メモリ制御手段にも、分散共有メモリアドレスを共有する送信プロセッサモジュール識別番号を記憶しておき、受信側分散共有メモリのメッセージ制御エリアにメッセージ制御情報を書き込むと、送信側分散共有メモリにコピーされる点が付加されている。また、分散共有メモリの処理要求エリアをFIFOメモリにする点も新たに追加されている。さらに、分散共有メモリ制御手段に登録されるプロセッサモジュール情報に応じて、一方方向性通信、両方向性通信、放送型通信等の各種の通信パターンを任意に実現できる点でも追加されている。このように、本発明によれば、メッセージが送信側から受信側に伝達されるまでの遅延時間が大幅に削減されるとともに、一方方向性、両方向性、放送型等の各種通信パターンを選択することができるので、柔軟性が高い。また、受信側プロセッサモジュールは受信側分散共有メモリの同じアドレスロケーションから制御データを読み出すだけで、プロセッサモジュール間の通信の同期を簡単にとることができる。かつ複数の送信プロセッサモジュールが同時に、分散共有メモリの同一番地に書き込みされた場合でも、受信側の分散共有メモリのFIFOに順次書き込まれるので、競合処理を全く必要としない。従って、受信側は、分散共有メモリ内の要求登録FIFOを順次読み出すのみで、発生した要求のみを効率よく取り出すことができ、従来のようなサーチのための処理オーバーヘッドを大幅に削減できる。

【実施例】以下、本発明の実施例を、図面により詳細に説明する。図4は、本発明の一実施例を示すマルチプロセッサシステムの構成図であって、3台のプロセッサからなる場合を示している。図4において、17はマルチプロセッサシステム、18-1、18-2、18-3はマルチプロセッサシステムにおけるプロセッサモジュール(PM)、19-1、19-2、19-3は各PM内のプロセッサ、20-1、20-2、20-3はそれぞれ対応するプロセッサ19-1~19-3から読み書きアクセスが可能なローカルメモリ、21-1、21-2、21-3は全プロセッサモジュール間で共通のアドレスが付与された分散共有メモリである。ただし、プロセッサ19-1からアクセスできる分散共有メモリはローカルメモリ21-1のみであって、他のPMのローカルメモリ21-2、21-3にはアクセスできない。同じように、プロセッサ19-2からアクセスできる分散共有メモリは21-2のみであり、プロセッサ19-3

12

からアクセスできる分散共有メモリは21-3のみである。22-1、22-2、22-3は分散メモリカップラであって、これらは分散共有メモリ21-1~21-3にデータがそれぞれ書き込まれた時点で、その書き込みデータを、予め指定された他のプロセッサモジュールの全てないし1個に送信すると同時に、他のプロセッサモジュールから受信した書き込みデータを、分散共有メモリ21-1~21-3のうちの送信側と同一のアドレスロケーションに書き込むための装置である。

【0013】次に、23-1、23-2、23-3は分散メモリプロテクタであって、それぞれ、分散共有メモリ21-1~21-3内のメッセージバッファに記憶されているメッセージを不正なアクセスから保護する装置である。24-1、24-2、24-3はネットワークアダプタであって、通信ネットワーク26を介して他の分散システムとメッセージ等を交換するための装置である。25はプロセッサモジュール間通信路であって、マルチプロセッサシステム17内のPM間でメッセージあるいはそれらの制御情報を転送するための装置である。プロセッサモジュール間通信路の具体的実現手段としては、システムバス、LAN、パケットスイッチングネットワーク、リングネットワーク等、送信側と受信側で1対1の通信ができるものであれば何でもよい。26は複数のマルチプロセッサシステム17間を相互接続するための通信ネットワークであって、ここでは転送データを固定長のブロックに分割して、それに行き先ヘッダを付加した53バイトのセルにして、ネットワーク内を転送するATM(Asynchronous Transfer Mode)ネットワークを想定しているが、その他の種類の通信ネットワークでも適用可能であることは勿論である。

【0014】図5、図6および図7は、図4における分散共有メモリのデータ配置例を示す図である。ここでは、データの配置を物理アドレスレベルで示している。分散共有メモリ内の通信領域は、①PM間通信エリア、②PM内通信エリア、③FIFO通信エリア、の3種類に大別される。このうち①PM間通信エリアは、マルチプロセッサシステム17内のPM間通信情報を記憶するエリアであり、②PM内通信エリアは、同一PM内で送受信される通信情報を記憶するエリアであり、また③FIFO(First In First Out)通信エリアは、PM間にまたがる処理要求をFIFO形式で蓄積記憶するエリアである。PM間通信エリアおよびPM内通信エリアは、

(i) メッセージを記憶するメッセージバッファ(MB)エリア

(ii) メッセージバッファの捕捉/解放状態を表示するMB管理マップエリア

(iii) メッセージバッファアドレス等の送受間引き継ぎ制御情報(ディスクリプタ)を複数個、連続アドレスに配置したディスクリプタリングエリア

13

それぞれに分けられる。これらのエリアの具体的使用方法については、追って説明する。上記MB、MB管理マップ、およびディスクリプタリングは、それぞれページを単位として物理メモリの割り当てが行われ、これらのページは仮想記憶のページテーブルにより管理されている。そして、MBエリア用のページテーブルには、プロセッサのスーパーバイザモード（カーネルやカーネルから起動される各種のシステム制御プログラムが実行されるモード）と、ユーザモード（各種アプリケーションプログラムが実行されるモード）との両方からアクセスが許されるように保護情報が設定されている。その結果、各種アプリケーションプログラムがMBエリアにメッセージを直接読み書きすることができるので、処理の効率化を図ることが可能となる。一方、MB管理マップ用ページと、MBディスクリプタリング用ページのページテーブルには、プロセッサのスーパーバイザモードのアクセスのみを許すように保護情報が設定されている。その結果、アプリケーションプログラムからMB管理マップおよびMBディスクリプタリングへの不正書き込みに対して保護が行われる。

【0015】図5～図7のエリア内の記述に示すように、分散共有メモリ21上の各エリアを識別するために、 M_{ij-k} のような識別名を用いている。ここで、Mはプロセッサモジュール対応のMB/MB管理マップ/MBディスクリプタリング/FIFO通信エリアの識別記号であり、PM(18-1)に対応するMBはX、そのMB管理マップはXM、MBディスクリプタリングはXDと表現される。同じように、PM(18-2)に対応するMBはY、そのMB管理マップはYM、MBディスクリプタリングはYD、FIFO通信エリアはYF、PM(18-3)に対応するMBはZ、そのMB管理マップはZM、MBディスクリプタリングはZD、FIFO通信エリアはZFのように名称が付けられている。また、 i, j は、それぞれ送信PMの識別番号(PM ID)と受信PMの識別番号(PM ID)を表わしており、PM#1(18-1)、PM#2(18-2)、PM#3(18-3)に対応してそれぞれ1, 2, 3の番号が付与されている。ただし、例外として、FIFO通信エリアXF, YF, ZFは値jのみを有し、値iを持っていない。その理由としては、FIFO通信エリアが値jで表わされる受信側プロセッサモジュール対応に分割されており、値iで表わされる各送信プロセッサモジュール間で共通に使用するため、エリアの指定情報として値iを必要としないためである。

【0016】kは同一種類のエリア内での複数個のMB/MB管理マップ/MBディスクリプタ/FIFO通信エリアを識別するためのもので、1から順に値が割り振られている。例えば、Z13-2は、分散共有メモリ21-3上の、PM#1(18-1)からPM#3(18-3)へのMBのうちの第2番目のMBを表わす。ま

14

た、XD21-2は、分散共有メモリ21-1上の、PM#2(18-2)から1番目のPM#1(18-1)へのMBディスクリプタのうち、第2番目のMBディスクリプタを表わす。図5～図7から明らかなように、PM間通信エリアでは $i \neq j$ 、PM内通信エリアでは $i = j$ となる。ただし、例外として、MBディスクリプタ/FIFO通信エリアで、送信処理要求情報が登録される場合にはSと表現され、受信処理要求情報が登録される場合にはRと表現される。例えば、XD12-Rは、分散共有メモリ(21-1)上のPM#1(18-1)からPM#2(18-2)への受信ディスクリプタを表わし、ZF2-Sは、分散共有メモリ(21-3)上のPM#2(18-2)への受信処理FIFO通信エリアを表わしている。各分散共有メモリは、実効的に必要とされるメモリ量、つまり自プロセッサが送受信するために必要なエリアのみが実装されている。例えば、分散共有メモリ21-1では、図5、図6のハッチ部分のみが実装されており、それ以外のエリアにはメモリが実装されていない。

【0017】2つのメッセージバッファの i, j, k の値がそれぞれ一致し、Mの値が異なるMBがペアを構成する。このペアは、異なる分散共有メモリの同一物理アドレスロケーションに配置される。例えば、図5～図7において、メッセージバッファX12-1とY12-1はペアを組んでおり、互いに同じ物理アドレスを有している。各ペアは、送信側PMのカーネルにより動的に捕捉、解放が行われる。各MBは固定長の大きさで構成されており、図5～図7に示すように、宛先PM-IDが同一の複数のMBを、分散共有メモリの連続するアドレスロケーションに割り付ける。これを、宛先PM対応MBプールと呼ぶ。各MBプール対応にMB管理マップが存在し、MBプール内のMB数に応じたMB管理マップのエントリが用意され、それらが分散共有メモリの連続エリアに割り付けられている。例えば、図5において、メッセージバッファX12-1, X12-2は1つのMBプールを構成し、X12-1に対してMB管理マップエントリXM12-1が、X12-2に対してMB管理マップエントリXM12-2が対応する。【対応】という意味は、MBの先頭アドレス（以下、単にMBアドレスと呼ぶ）が付与されると、そのMB管理マップエントリアドレスを求めることができ、逆にMB管理マップエントリアドレスからMBアドレスも求めることができることを意味している。MB管理マップの各エントリは、対応するMBが「未使用」か「使用中」かの状態を表示し、カーネルにより値が設定される。MBの場合と同じ考え方により、送信側分散共有メモリMB管理マップエントリと、受信側分散共有メモリの同一アドレスのMB管理マップエントリがペアを構成している。例えば、MB管理マップエントリXM12-2とYM12-2がペアを組んでいる。

15

【0018】 ディスクリプタリングエリアも、MBと同じように、宛先PM-IDが同一の複数のMBディスクリプタを分散共有メモリの連続するアドレスロケーションに割り付けており、割り付けた全体をディスクリプタリングと呼ぶ。『リング』と呼ばれている理由は、ディスクリプタリングに含まれる複数のディスクリプタを若いアドレス順にサイクリックに使用していくからである。送信側の各ディスクリプタリングと同一アドレスのエリアが、受信プロセッサモジュールの分散共有メモリにも配置される。ディスクリプタリングは、送信オブジェクトからのメッセージ制御情報を他のPMあるいは他のシステムに引き継ぐための『送信ディスクリプタリング』と、他のPMあるいは他のシステムから受信したメッセージ制御情報を受信オブジェクトに引き継ぐための『受信ディスクリプタリング』に分けられる。ディスクリプタリングもMBやMB管理マップの場合と同じように、送信側と受信側の同一アドレス間でペアを構成する。ディスクリプタリング内のディスクリプタとMBとの対応は、実行時にカーネルにより動的に決定される。具体的には、例えばPM#1(18-1)からPM#2(18-2)にメッセージを送信する場合に、PM(18-1)のカーネルが空きのMBプールの中から適当なMB(例えばX12-2)を捕捉し、次に対応する送信ディスクリプタリング(例えばXD12-S)を選択し、その中の『未使用』の最若アドレスのディスクリプタにMBアドレスを登録する。このような方法でディスクリプタとMBとの対応が動的に決定される。なお、送信ディスクリプタリングXD12-Sとペアを組む送信ディスクリプタリングはYD12-Sである。

【0019】 図7におけるFIFO通信エリアは、送信オブジェクトからのメッセージ送信処理要求を他のPMまたは他のシステムに通知するための『送信処理FIFOエリア』と、他のPMまたは他のシステムから受信したメッセージ受信処理要求を受信オブジェクトに通知するための『受信処理FIFOエリア』に分けられる。このうち、メッセージ処理要求送信側の分散共有メモリは、通常のRAM(ランダムアクセスメモリ)で構成されるが、メッセージ処理要求受信側の分散共有メモリは、FIFOで構成されている。例えば、PM#1(18-1)上のPM#1(18-1)宛受信処理FIFOエリアXF1-Rは、他のPMからのメッセージ受信処理要求を記憶するためにFIFOメモリで構成されているが、PM#2上のPM#1(18-1)宛受信処理FIFOエリアYF1-Rおよび、PM#3上のPM#1(18-1)宛受信処理FIFOエリアZF1-Rは、いずれもRAMで構成されている。この理由としては、受信処理要求を書き込む側のPM(PM#2とPM#3)が受信処理要求をそれぞれ受信処理FIFOエリアYF1-RとZF1-Rに同時に書き込んだ場合、それらが同時に受信処理要求を読み出す側のPM(PM#

16

1)の受信処理FIFOエリアXF1-Rに到着するので、これらの受信処理要求を全て蓄積するためにFIFOメモリを使用するからである。

【0020】 図1は、本発明の一実施例を示すプロセッサモジュールの内部構成図である。図1のプロセッサモジュールは、図4におけるPM18-1~18-3に該当している。なお、以下の説明では、3つのPM内に存在する構成要素の各々を識別する必要がないときには、-1、-2、-3の各識別符号を省略する。例えば、個々の分散共有メモリを指定しないときには、分散共有メモリ21-1、21-2、21-3と記述することなく、単に分散共有メモリ21と記述することにする。図1において、16は分散共有メモリ21上に配置されたメッセージバッファ(MB)、30はプロセッサモジュール(PM)18内の各種装置を結合するためのプロセッサバス、22は分散メモリカップラで、分散共有メモリ21に実装されているエリアへの書き込みアクセスがあったとき、他のPMの分散共有メモリの同じアドレスロケーションにも書き替えデータを送信する機能、および他のPMから書き替えデータを受信して、自PMの分散共有メモリへの書き込みを行う機能を有する。分散メモリカップラ22は、以下の構成要素を含んでいる。先ずバス信号デコーダ31であり、これはプロセッサバス30の信号線上の信号を解釈し、所属する分散共有メモリ21への書き込みアクセスがあれば、信号線32に‘1’を出力して、転送制御部33を起動する。転送制御部33は、分散メモリカップラ22全体の制御を司る部分であり、ここから内部ロジックに各種制御信号(図示省略)を供給する。34はモジュールID管理部であり、分散共有メモリ21への書き込みデータをPM間で送受する際に必要な宛先プロセッサモジュールID(PM ID)を記憶している。モジュールID管理部34の詳細については図6に示している。

【0021】 図1において、35はパケット送信レジスタであって、他のPMに転送する分散共有メモリアドレスや書き込みデータ等、または他のPMへの割り込み情報等を記憶する。36はパケット送信バッファであり、パケット送信レジスタ35からのデータを受け取り、他のPMに転送するまでの一時待ち合わせ機能を有している。37はパケット受信バッファであり、他のPMから転送されてきたデータを受け取り、パケット受信レジスタ38に渡すまでの一時待ち合わせの機能を有する。パケット受信レジスタ38は、他のPMから転送されてきた情報を記憶するレジスタであり、パケット送信レジスタ35が送り出した内容と同じデータが記憶される。39はパケットデコーダであり、パケット受信レジスタ38内のデータをデコードして、分散共有メモリ21内のアドレスロケーションへの書き込み要求であれば、信号線40を介して転送制御部33に分散共有メモリ書き込みの実行を依頼する。また、デコードの結果が、他のP

17

Mからの割り込みであれば、信号線41を介して割り込み制御部42に割り込み処理を依頼する。割り込み制御部42は、信号線43を経由してプロセッサ19に割り込み要求を行う。

【0022】図1において、23は分散メモリプロテクタであって、分散共有メモリ21に配置されたMB(16)内のメッセージを不正なアクセスから保護するハードウェア機構である。分散メモリプロテクタ23は、カレントケーバリティレジスタ(CCR: Curent Capability Register)51、MCRメモリ(Memory Capability Register Memory)50、比較器52を含んでいる。CCR51は、カレントケーバリティを記憶するレジスタである。カレントケーバリティは、具体的には、プロセッサ19で実行中のアプリケーションオブジェクト(プロセッサのユーザモードで実行されるオブジェクト)のIDを意味している。MCRメモリ50は、複数のMCR53から構成され、各MCR53は分散共有メモリ21上の各MB16対応に、メモリーケーバリティを記憶する。メモリーケーバリティは、具体的には、対応するMB16へのメモリアクセスが生じると、MBアドレスを基に対応するMCR53をMCRメモリ50の中から選択し、これを取り出す。比較器52は、選択されたMCR53とCCR51とを比較してメッセージバッファへの正しいアクセスであるか否かを判定し、不一致であれば不正アクセスであると判定する。不正アクセスを検出した場合には、並行して行われているMB16へのアクセスを中継し、信号線54を介してプロセッサ19に緊急��通する。

【0023】以下、分散メモリプロテクタ23を、PM18-1内の送信オブジェクトからPM18-2内の受信オブジェクトへのメッセージ通信に適用した場合の制御手順について述べる。

(a) 送信PM18-1のカーネルは、送信オブジェクトの実行を開始する前に、カレントケーバリティレジスタCCR51-1にカレントケーバリティつまり送信オブジェクトIDを設定する。

(b) 送信PM18-1のカーネルは、分散共有メモリ21-1上に送信MB16-1を捕捉した後、送信MB16-1対応のMCR53-1にメモリーケーバリティを設定する。この時点のメモリーケーバリティは、MBの捕捉を要求したオブジェクトのIDつまり送信オブジェクトIDである。

(c) 送信プロセッサ19-1から送信MB16-1へのメモリアクセスがある度毎に、MCR53-1の中からMBアドレスを基に対応するMCR53-1を1個選択して、それをCCR51-1と比較する。MCR53-1とCCR51-1のオブジェクトIDの値が等しいときには正しいアクセスであり、等しくないときには不正アクセスであると判定する。

(d) メッセージが受信PM18-2の受信MB16-

18

2に到着したと仮定する。受信PM18-2のカーネルは、受信オブジェクトの実行を開始する前に、CCR51-2にカレントケーバリティつまり受信オブジェクトIDを設定する。

(e) 受信PM18-2のカーネルは、受信MB16-2に対応するMCR53-2に、メモリーケーバリティを設定する。この時点のメモリーケーバリティは、メッセージの宛先であるオブジェクトのIDつまり受信オブジェクトIDである。

【0024】(f) 受信プロセッサ19-2から受信MB16-2へのメモリアクセスがある度毎に、比較器52-2によりMCR53-2とカレントケーバリティレジスタCCR51-2内のオブジェクトIDの値を比較し、等しければ正しいアクセス、等しくなければ不正アクセスと判定する。

上記動作において、MCR53とCCR51に、それぞれメモリーケーバリティとカレントケーバリティが設定されている期間(プロテクションウィンドウ開放期間)は、MBの所有権が付与された送信または受信オブジェクトのみがそのMBをアクセスできる。プロテクションウィンドウ開放期間中に、もし無関係のオブジェクトがMBアクセスを行うと、MCRとCCRのケーバリティの不一致が生じて、不正アクセスであることが検出される。プロテクションウィンドウ開放期間以外では、送信、受信オブジェクトも含めた全てのアプリケーションオブジェクトは、このMBにアクセスすることができないので、強固なメモリ保護が実現されることになる。なお、分散メモリプロテクタ23は、プロセッサのアプリケーションモード(またはユーザモード)でのMBアクセスで動作するが、カーネルモード(またはスーパーバイザモード)でのメモリアクセスは動作せず、無条件にメモリアクセスを許容する。この理由としては、一般にカーネルモードで実行されるプログラム(例えばカーネル)は、アプリケーションオブジェクトに比較して潜在バグの含有率が小さく、不正なメモリアクセスをする可能性が少ないからである。

【0025】図8は、図1におけるモジュールID管理部の内部構成図である。図8において、60はモジュールIDディレクトリであり、CAM(Content Addressable Memory)セル部61とデータメモリ部62から構成される。CAMとは、検索キーデータを入力して、これと各ワード(CAMセル)の記憶データの内容を一斉に比較照合して、指定された検索条件に合致した内容のCAMセルを選択表示するメモリである。CAMセルの各々には、「分散共有メモリページアドレス63」が記憶されている。分散共有メモリページアドレス63は、他のプロセッサモジュールと共用されているアドレスのページである。例えば、分散共有メモリ21の連続エリアに配置された宛先PM対応のMBブールのページアドレス、そのMB管理マップのページアドレス、ディスク

19

リプタリングのページアドレス、宛先PM対応のFIFO通信エリアのページアドレス等が設定される。図5～図7で説明したように、MB、MB管理マップ、ディスクリプタリング、FIFO通信エリアは、それぞれ宛先PM毎に異なるページに割り付けられている。一方、データメモリ部62は、各CAMセルのワードに対応する補助データを記憶するRAMであり、分散共有メモリページアドレス63を共有している宛先プロセッサモジュールのID（宛先PM ID）64を記憶している。

【0026】ここで、MB管理マップ、ディスクリプタリングは、送信PM、受信PMの両方から書き込みが行えるようにするため、送信PMのモジュールIDディレクトリ60には、〔共有ページアドレス、受信PM ID〕のペアが記憶され、受信PMMのモジュールIDディレクトリ60には、〔送信側と同じ共有ページアドレス、送信PM ID〕のペアが記憶される。このように、各PMのモジュールIDディレクトリ60には、コピーをしたい全てのPMのPM IDが登録されている。図8の65はキーレジスタであり、これはCAMセル部61の内容と比較照合するための検索キーデータを記憶する。66はマスクレジスタであり、これはキーレジスタ65内の検索キーデータとCAMセル部61との比較を行う場合に、比較しないビット位置を指定するためのレジスタである。67はCAMコマンドレジスタであり、CAMセル部61に関する各種指令を記憶する。68は結果レジスタであり、これは検索した結果、附番マCAMセルの有無の表示、複数セル一致の有無の表示、空きセルの有無の表示等を行う部分である。キーレジスタ65とCAMセル部61との比較処合の結果、一致したCAMを使用しなくても、よく知られているようにハッシングの技術とRAMとを組み合わせ、CAMと同じような機能を実現することができる。

【0027】図8に示すモジュールID管理部34は、以下のように動作する。分散共有メモリ21への書き込みアクセスがあると、そのアドレスがキーレジスタ65に格納され、そのページアドレスとCAMセル部61とが比較される。一致したセルが検出されると、そのアドレスが他のPMにより共用されていることを意味するので、データメモリ部62から対応する宛先PM ID64を読み出し、これを分散共有メモリアドレスに付与した後、書き込みデータと合わせて送信パケットを作成し、図1に示すパケット送信レジスタ35に設定する。パケット送信レジスタ35からパケット送信バッファ36を経由してパケットを宛先PMに送出する。受信側の分散メモリカップラ22では、パケット受信バッファ37で、他のPMから転送されてきたデータを受け取り、パケット受信レジスタ38に引き渡す。パケット受信レジスタ38では、受信したデータをデコードし、指定された分散共有メモリアドレスロケーションにデータを書き込む。このようにして、送信側分散共有メモリから受

20

信側分散共有メモリにコピーが行われる。なお、送信側分散共有メモリへの書き込みアクセスでは、受信側分散共有メモリへのコピーアクセスの完了を待たずにアクセスを完了する。

【0028】本実施例においては、あるPMから他のPMにポイントツーポイントでコピーを行う構成を基本として説明しているが、ポイントツーマルチポイントでコピーを行うことも可能である。その場合には、モジュールIDディレクトリ60には1つの分散共有メモリページアドレス63に対して、複数の宛先PM ID64を登録しておく。その結果、分散共有メモリへの1回の書き込みに対して複数のパケットが生成され、それぞれ指定されたPMにパケットが送信されるので、複数のPMの分散共有メモリにコピーを行うことができる。また、同一の分散共有メモリページアドレス63に対して、送信側PMのモジュールIDディレクトリ60には、受信側PM IDを登録し、受信側PMのモジュールIDディレクトリ60には、送信側PM IDを登録しておくことにより、分散共有メモリのエリアに送信側、受信側のどちらから書き込んでも、互いに相手側の分散共有メモリにコピーを行う、つまり双方向通信を行うことができる。

【0029】図9は、図1におけるパケット送信レジスタおよびパケット受信レジスタに共通のデータフォーマット図である。ここでは、分散共有メモリアクセスタイプと割込みデータタイプの2種類のフォーマットを示している。図9において、80、90は属性識別フィールドであり、‘0’ならば分散共有メモリアクセスタイプであることを示し、‘1’ならば割込みデータタイプであることを示す。分散共有メモリアクセスタイプの場合には、モジュールID管理部34が出力する『宛先PM ID81』、『分散共有メモリアドレス82』、『書き込みデータ83』、『書き込み単位84』の各フィールドを含む。『書き込み単位84』フィールドは、書き込みのデータ幅、例えばバイト、ハーフワード（16ビット）、ロングワード（32ビット）等を指定するものである。一方、割込みデータタイプの場合には、モジュールID管理部34が出力する『宛先PM ID91』、割込み発生源を示す『送信モジュールID92』、割込みの種別を示す『割込み識別データ93』、『書き込み単位94』の各フィールドを含む。なお、宛先PM ID81、91がある特定の値の場合には、放送型パケットとして全てのPMでパケットを受信するように構成することが可能である。

【0030】図10、図11、図12は、本発明において使用される各種のメッセージ通信のパターンと、その処理の流れを示す説明図である。図10はシステム内メッセージ通信の場合である。図10において、120は送信オブジェクト、121は受信オブジェクト、122はメッセージキュー、123は送信オブジェクト、12

21

4は受信ディスクリプタリング、125はIOC配信オブジェクト、126はメッセージキュー、127は受信オブジェクトである。メッセージパターンは、以下のよう分類される。

(a) システム内メッセージ通信・・・送信オブジェクトと受信オブジェクトが同一マルチプロセッサシステム(図1の17)内に存在する場合である。システム内メッセージ通信には、次の2通りのケースが存在する。

(a1) PM内通信・・・送信オブジェクト120と受信オブジェクト121が同一PM内に存在する場合である。PM内通信の場合、カーネルは受信オブジェクト121のメッセージキュー122のアドレスを知っているため、カーネルは、送信オブジェクト120から受信オブジェクト121のメッセージキュー122に直接、メッセージを登録することができる。受信オブジェクト127はメッセージキュー126からメッセージを取り出し、対応する処理を行う。

(a2) PM間通信・・・送信オブジェクト123と受信オブジェクト127が同一マルチプロセッサシステム17内の異なるPMに存在する場合である。PM間通信の場合、送信オブジェクト123の存在するPM(送信PM)のカーネルからは、受信PM内の受信オブジェクト127の制御情報(例えばメッセージキュー126のアドレス)が見えないので、直接、受信オブジェクト127のメッセージキュー126にメッセージを登録することができない。そこで、送信PMでは、送信カーネルが、事前に知っている受信ディスクリプタリング124にメッセージを登録し、受信PMではIOC(Interobject Communication)配信オブジェクト125が、受信ディスクリプタリング124を読み出し、受信オブジェクト127のメッセージキュー126にメッセージを登録する。

【0031】図11は、システム間メッセージ通信の場合である。図11において、140は送信オブジェクト、141は送信ディスクリプタリング、142はINC送信処理オブジェクト、143はアダプタ送信ディスクリプタリング、144はアダプタ受信ディスクリプタリング、146はメッセージキュー、147は受信オブジェクト、150は送信オブジェクト、151は送信ディスクリプタリング、152はINC送信処理オブジェクト、153はアダプタ送信ディスクリプタリング、154はアダプタ受信ディスクリプタリング、155はINC受信処理オブジェクト、156は受信ディスクリプタリング、157はIOC配信オブジェクト、158はメッセージキュー、159は受信オブジェクトである。

(b) システム間メッセージ通信・・・送信オブジェクトと受信オブジェクトが異なるマルチプロセッサシステム17に存在する場合である。送信システムあるいは受信システムで中継PMがない場合と、ある場合の2通りがある。

22

(b1) 中継PMがない場合・・・送信システムの場合、送信オブジェクト140の存在するPM(送信PM)と、受信システム宛の通信リンクを持っているPM(送信中継PM)とが一致している場合である。受信側システムの場合、通信リンクを終端しているPM(受信中継PM)と受信オブジェクト147の存在するPM(受信PM)とが一致している場合である。送信システムで送信中継PMがない場合、送信PMでは、カーネルが送信オブジェクト140からのメッセージを自PM内の送信ディスクリプタリング141に登録する。INC(Inter-System Communication)送信処理オブジェクト142は、送信ディスクリプタリング141を読み出してネットワークプロトコル処理を行い、アダプタ送信ディスクリプタリング143にメッセージを登録する。ネットワークアダプタ24は、アダプタ送信ディスクリプタリング143からメッセージを取り出して、通信ネットワーク26に送出する。

【0032】一方、受信システムで受信中継PMがない場合、通信ネットワーク26からメッセージが送り届けられ、受信PMのネットワークアダプタ24がメッセージをアダプタ受信ディスクリプタリング144に登録する。そして、INC受信処理オブジェクト145がアダプタ用受信ディスクリプタリング144を読み出して、同一PM内の受信オブジェクト147のメッセージキュー146にメッセージを登録する。

(b2) 中継PMがある場合・・・送信システムの場合、送信オブジェクト150の存在するPM(送信PM)が、受信システムへの通信リンクを持ち合わせておらず、他のPM(送信中継PM)が持ち合わせている場合である。また、受信システムの場合、通信リンクを終端しているPM(受信中継PM)と受信オブジェクト159の存在するPM(受信PM)とが異なる場合である。送信システムでは、送信中継PMがある場合、送信PMのカーネルの送信処理プログラム17が送信オブジェクト150からのメッセージを送信中継PM宛の送信ディスクリプタリング151に登録する。送信中継PMのINC送信処理オブジェクト152は送信ディスクリプタリング151を読み出して、ネットワークプロトコル処理を行ってアダプタ送信ディスクリプタリング153にメッセージを登録する。ネットワークアダプタ24はアダプタ送信ディスクリプタリング153からメッセージを取り出して通信ネットワーク26に送出する。

【0033】また、受信システムで受信中継PMがある場合、通信ネットワーク26からメッセージが送り届けられ、受信PMのネットワークアダプタ24がメッセージをアダプタ受信ディスクリプタリング154に登録する。次に、受信中継PMのINC受信処理オブジェクト155がアダプタ受信ディスクリプタリング154を読み出し、最終宛先の受信PM宛の受信ディスクリプタリング156にメッセージを登録し、受信PMではIOC

23

(Interobject Communication) 配信オブジェクト 157 が受信ディスクリプタリング 156 を読み出し、受信オブジェクト 159 のメッセージキュー 158 にメッセージを登録する。なお、図 11 のシステム間メッセージ通信の組み合わせパターン数は、〔送信中継あり／なし〕, 〔受信の中継あり／なし〕の合計 4 通りである。図 12 は、各アプリケーションオブジェクトに付与されたオブジェクト ID のフォーマットを示す図である。図 12 において、170 はオブジェクト ID、171 はそのオブジェクトが存在するマルチプロセッサシステムの番号で、通信ネットワーク内で一意に識別可能なように形成される。172 はプロセッサモジュール ID (PM ID) で、マルチプロセッサシステム内で一意に識別可能な値をとる。173 はローカル ID であり、PM 内でオブジェクトを一意に識別するための番号である。

【0034】図 13 は、本発明における送信ディスクリプタと送信処理要求のメモリ配置例を示す図であり、図 14 は、送信処理要求のフォーマット図である。PM# 1 (18-1) から PM# 3 (18-3) 宛の送信ディスクリプタリングを (XD13-S, ZD13-S) のペアで、PM# 2 (18-2) から PM# 3 (18-3) 宛の送信ディスクリプタリングを (YD23-S, ZD23-S) のペアで、PM# 3 内の送信ディスクリプタリングを ZD33-S で、それぞれ表わす。また、PM# 1 (18-1) 上の PM# 3 (18-3) 宛の送信処理 FIFO を XF3-S で、PM# 2 (18-2) 上の PM# 3 (18-3) 宛の送信処理 FIFO を YF3-S で、PM# 3 (18-3) 内の送信処理 FIFO を ZF3-S で、それぞれ表わす。このように、図 13 はシステム間 PM 中継通信の送信システムの例を示している。送信オブジェクトは PM# 1 (18-1)、PM# 2 (18-2)、PM# 3 (18-3) に分散しており、送信中継 PM は PM# 3 (18-3) とする。各 PM の送信オブジェクトから合計 5 個のメッセージ M_i ($i=1, 2, \dots, 5$) が PM# 3 (18-3) を経由して通信ネットワークに送出されるものとし、各メッセージの送信ディスクリプタを順に Q1, Q2, Q3, Q4, Q5 と表わし、それらに対応する送信処理要求を順に QA1, QA2, QA3, QA4, QA5 と表わす。送信ディスクリプタの構造は、後述の図 15 (c) に示されている。送信処理要求のデータ構造 (フォーマット) は、図 14 に示すように、要求元プロセッサモジュール ID (PM ID) 175 と、送信ディスクリプタへのポインタ 176 から構成されている。送信要求は、次のように伝達される。まず、PM# 1 (18-1) でメッセージ M_1 の送信要求が発生すると、PM# 1 (18-1) では PM# 3 (18-3) 宛送信ディスクリプタリング XD13-S に送信ディスクリプタ Q1 を登録し、次に PM# 3 宛送信処理 FIFO エリア XF3-S に送信処理要求 QA1 を登録する。これらが分散メモリ

24

カップラ 22 により、それぞれ PM# 3 上の送信ディスクリプタリング ZD13-S と PM# 3 上の送信処理要求 FIFO エリア ZF3-S にコピーされる。

【0035】次に、2 番目のメッセージ M_2 の送信要求が PM# 3 (18-3) 内で発生したと仮定すると、PM# 3 (18-3) 内の送信 MB ディスクリプタリング ZD33-S に送信ディスクリプタ Q2 を、PM# 3 (18-3) 宛送信処理 FIFO エリア ZF3-S に送信処理要求 QA2 を登録する。この場合には、PM 内通信であるため、他の PM へのコピーは行われない。なお、送信処理 FIFO エリア ZF3-S は FIFO メモリ構造であるため、最初に登録された QA1 が保存されたまま QA2 が追加登録される。次に、3 番目のメッセージ M_3 の送信要求が PM# 2 (18-2) で発生したものとすると、PM# 2 (18-2) 上の PM# 3 宛送信 MB ディスクリプタリング YD23-S に送信ディスクリプタ Q3 が PM# 3 宛送信処理 FIFO エリア YF3 に QA3 が登録される。それらが、それぞれ PM# 3 (18-3) 上の送信 MB ディスクリプタリング ZD23-S と PM# 3 上の送信処理要求 FIFO エリア ZF3-S にコピーされる。送信処理 FIFO エリア ZF3-S には QA1, QA2 が保存されたまま、QA3 が追加登録される。送信ディスクリプタ Q4, Q5 についても、全く同じように、Q5 までの要求登録が終了した段階では、図 13 のような記憶状態となる。

【0036】この段階において、PM# 3 (18-3) の INC 送信処理オブジェクトは、PM# 3 (18-3) 宛の送信処理 FIFO エリア ZF3-S を読み出すと、入力した順に送信処理要求 QA1, QA2, QA3, QA4, QA5 を取り出すことができる。要求の中に記載されている送信元 PM ID 175 と送信ディスクリプタポインタ 176 から、対応する送信ディスクリプタ Q_i の所在位置を求め、送信ディスクリプタ Q_i からメッセージバッファの所在位置を求めることができる。このようにして、メッセージ M_i を順に処理してネットワークに送出していく。ここで、従来のポーリング方式では、3 つの送信ディスクリプタリング ZD13-S, ZD23-S, ZD33-S を順にポーリングして送信要求の有無をチェックしており、処理要求の有無にかかわらず、全ての送信ディスクリプタリングを見る必要があった。そのため、読み出しても送信要求が見つからない空振りが生じることがあり、その結果、処理のオーバーヘッドが大きかった。これに対して本実施例では、PM# 3 (18-3) で自 PM 宛送信処理 FIFO エリア ZF3-S を周期的に読み出すだけで発生した送信要求を取り出すことができ、無駄な空振りが生じることがないので、極めて処理効率が高い。また、PM# 1 (18-1) から送信処理 FIFO エリア XF3-S を経由した送信処理 FIFO エリア ZF3-S への登録と、PM# 2 (18-2) から送信処理 FIFO エリア YF3

25

-Sを経由した送信処理FIFOエリアZF3-Sへの登録が同時に発生しても、送信処理FIFOエリアZF3-SのFIFOメモリの構造により、登録要求が直列化されて全て蓄積されていくため、マルチプロセッサ間の競合処理を行う必要がなく、処理オーバーヘッドは極めて軽減される。なお、図13では、送信要求の場合を例に説明しているが、受信要求の場合にも全く同じように、自PM宛の受信要求を指定された受信処理FIFOエリアから読み出すため、発生した受信要求のみを効率よく検出できる。

【0037】図15および図16は、本発明で使用される各種データ構造を示す図である。図15(a)はメッセージバッファ(MB)180のデータ構造であり、NEXTMP181は受信オブジェクト宛のメッセージをリスト構造で接続するためのポインタ、SID182はこのメッセージの送信元である送信オブジェクトのID、RID183はこのメッセージの宛先である受信オブジェクトのID、SIZE184はメッセージ本体(BODY)186のサイズを表わしている。また、ATTR185はこのメッセージの付属属性を表示したフィールド、BODY186はメッセージの中味を示すフィールドである。MB180の先頭アドレスは、MBA202として付与される。各MBは固定長であり、前述のように宛先PM IDが同じであるMB群が分散共有メモリ21の連続するエリアに割り付けられて1つのMBプールを構成している。各MBプール対応にMB管理マップが存在する。このMB管理マップは、MBプール内のMB数に応じたMB管理マップエントリから構成される。

【0038】図15(b)はMB管理マップエントリV190の構造を示しており、V=0の場合には、対応するMBが「未使用」の状態であり、V=1の場合には、対応するMBが「使用中」の状態を表わしている。MBが1つ与えられると、その先頭アドレスMBA200の値から、対応するMB管理マップエントリV190のアドレスが簡単に計算で求められるように構成される。同じように、各宛先PM対応MBプール毎に送信ディスクリプタリングと受信ディスクリプタリングが1つずつ存在して、両者を合わせてディスクリプタリングと呼ぶ。図14で説明したように、送信ディスクリプタリングは他のシステムへのメッセージ制御情報を記憶し、受信ディスクリプタリングは他のシステムから、あるいはシステム内の他PMからのメッセージ制御情報を記憶する。ディスクリプタリングの各エントリ(ディスクリプタ)はMBの各々に対応している。図15(c)は、ディスクリプタ200の構造を示したものであって、201は論理リンク番号で、対応するMB上のメッセージを通信ネットワーク26経由で他のシステムに転送する場合に使用する通信リンクの識別番号である。202は対応するMBの先頭アドレスMBAを記憶するフィールドで、

26

図15(a)で示したMBA202と同一である。203は実行権を表わす情報であり、実行権=「ENQ」であるならば、ディスクリプタリングにディスクリプタを登録できる状態(エンキュー)にあることを示しており、実行権=「DEQ」であれば、ディスクリプタリングからそのディスクリプタを取り出せる状態(デキュー)にあることを示している。

【0039】図15(d)はアプリケーションオブジェクトの実行に必要な各種の制御データを記憶するオブジェクトコントロールブロック(OCB)の構造を示したもので、OCBA211はOCBの先頭アドレスを示すポインタ、NEXTOP212はOCBをレディキューにリンクする時のポインタ、MSGP213はオブジェクト宛のメッセージを記憶しているMBのアドレスを示す。STATUS214は、オブジェクトの実行状態を示すフィールドであり、送受信オブジェクト間の同期制御に使用される。その他の制御215は、その他の制御フィールドである。図16では、プロセッサ19実行待ちのOCBを登録したレディキューの構造と、各種制御データ構造との関係を示している。READYP220で示されるレディキューに2つのOCB(OCBi221, OCBj222)が登録されており、OCBi221は2個のメッセージを受信し、OCBj222は3個のメッセージを受信している状態にある例を示している。

【0040】図17は、システムルーチングテーブルとPMルーチングテーブルの構造図である。受信オブジェクトID(RID)(図15の183)の値を基にして、メッセージをシステム内のどのPMを経由して送受信するかを指定するルーチングテーブルであって、本実施例では、システムルーチングテーブル249とPMルーチングテーブル289から構成され、それぞれ各PMのローカルメモリ20に記憶されている。システムルーチングテーブル249の各エントリは、システムIDフィールド250、システム通信モードフィールド251、通信リンクフィールド252、送信中継PMフィールド253から構成されている。システムIDフィールド250に「自システムID260」が登録されているエントリの場合には、そのエントリのシステム通信モードフィールド251は、「システム内通信261」と表示されている。一方、システムIDフィールド250に「他システムID270」が登録されているエントリの場合には、そのエントリのシステム通信モードフィールド251は「システム間通信271」と表示され、通信リンクフィールド252には、他システムに結合されている通信リンクの「通信リンクID272」が登録されている。また、送信中継PMフィールド253には、通信リンクID272に対応する通信リンクを収容している「送信中継PMのID273」を記憶している。あるマルチプロセッサシステム17が、他のK個のマルチ

27

ロセッサシステムと接続されている場合には、システムルーチングテーブル249のエントリ総数は $(1+K)$ 個となる。

【0041】PMルーチングテーブル289は、PM IDフィールド290、PM通信モードフィールド291、MB管理マップアドレスフィールド292から構成される。PM IDフィールド290に「自PM ID 300」が登録されているエントリの場合には、そのエントリのPM通信モードフィールド291には「PM内通信301」と記述されており、MB管理マップアドレスフィールド292には「PM内通信MB管理マップベースアドレス302」が登録されている。また、PM IDフィールド290に、「他PM ID 310」が登録されているエントリの場合には、PM通信モードフィールド291には「PM間通信311」と記述されており、MB管理マップアドレスフィールド292には、そのPMIDに対応する「PM間通信MB管理マップベースアドレス312」が登録されている。各フィールドの詳細な使用方法は後述する。なお、あるマルチプロセッサシステム17がJ個のPMから構成されている場合には、PMルーチングテーブルのエントリ総数はJ個である。

【0042】図18は、本発明におけるシステム内PM内通信のタイムチャートであり、図19は同じくシステム内PM間通信のタイムチャートであり、図20はシステム間PM無中継通信のタイムチャートであり、図21はシステム間PM中継通信のタイムチャートである。なお、図18～図21のプロセッサの実行ステップのうち、太い野線で示されたステップ（例えば、図18のステップS100）はプロセッサのスーパーバイザモードで実行している部分であり、これに対して細い野線で示されたステップ（例えば、図18のステップS102）はプロセッサのユーザモードで実行している部分である。図22～図32は、アプリケーション（APL）オブジェクトからの要求により、カーネル自身のプログラムあるいはカーネルが起動するプログラムのフローチャートであって、いずれも全てプロセッサのスーパーバイザモードで実行される。図22は、カーネルの一部であって、アプリケーション（APL）オブジェクトの起動および終了処理を行うプログラムのフローチャートである。また、図23および図24は、MB捕捉処理プログラムおよびMB解放処理プログラムの各フローチャートであって、APLオブジェクトからの要求により起動される。また、図25および図26は、送信処理および受信処理プログラムのフローチャートであり、これらもAPLオブジェクトからの要求により起動される。

【0043】図27および図28は、送信ディスクリプタリング、受信ディスクリプタリング、アダプタ送信ディスクリプタリング、アダプタ受信ディスクリプタリングに共通なエンキュー処理（各ディスクリプタリングに

28

エントリを登録する処理）およびデキュー処理（各ディスクリプタリングからエントリを削除する処理）のプログラムのフローチャートである。図27（a）はディスクリプタリングの一般的な論理構造を示す図、図27

（b）はエンキュー処理のフローチャート、図28はデキュー処理のフローチャートである。図27（a）において、400はリードポインタ（RP）であり、ディスクリプタリングからエントリを取り出す場合（デキュー処理）のエントリ読み出しアドレスを指定する。401はライトポインタ（WP）であり、ディスクリプタリングにエントリを登録する場合（エンキュー処理）のエントリ書き込みアドレスを指定する。各エントリは実行権フィールド403と、処理要求内容を記憶するディスクリプタアイテムフィールド404に分割される。ディスクリプタアイテムフィールド404は、ディスクリプタリングの種類（例えば、送信ディスクリプタリングか／アダプタ送信ディスクリプタリングかの違い）に依存して内容が変わることがある。実行権フィールド403は、そのエントリにアクセスする主体がエンキュー側にあるか／デキュー側にあるかを指定するもので、「ENQ」と表わされているときには新しいエントリを登録してもよいことを意味し、「DEQ」の表わされているときには登録されているエントリを取り出してもよいことを意味する。エンキュー側はエントリを登録する毎にWP401をインクリメントし、次のエンキュー位置を知らせる。また、デキュー側はエントリを取り出す毎にRP400をインクリメントし、次のデキュー位置を知らせる。

【0044】図27にエンキュー処理、図28にデキュー処理の各フローが示される。エンキュー処理では、ステップ420で、WP401で示されるエントリの実行権フィールド403が「ENQ」と表示されているか否かをテストする。もし、ENQと表示されているならば、新しいエントリを登録してもよいことを意味するので、ステップ421でそのディスクリプタアイテムフィールド404に新エントリ情報を書き込み、ステップ422でそのエントリの実行権フィールド403を「DEQ」に書き替える。ステップ423では、このエントリの登録によりディスクリプタリングの最後に到達したか否かをテストする。到達していなければ、ステップ424でWP401をインクリメントし、次のエンキュー位置を設定する。ディスクリプタリングの最後に到達していれば、ステップ425でWP401の値をディスクリプタリングのベースアドレスにセットすることにより、ディスクリプタリングをサイクリックに使用する。ステップ420で実行権フィールド＝「DEQ」と表示されているときには、これはディスクリプタリング402に前回登録したエントリがデキュー側により未だ読み出されずに残っていることを示すので、ステップ426でディスクリプタリングオーバフローのエラー処理を実行す

29

る。

【0045】一方、デキュー処理では、ステップ430でRP400で示されるエントリの実行権フィールド403が「DEQ」と表示されているか否かをテストする。もし、DEQと表示されていれば、エントリが登録されており、そのエントリを取り出してもよいことを意味するので、ステップ431でそのディスクリプタリングアイテムフィールド404からディスクリプタを取り出し、ステップ432でその実行権フィールド403を「ENQ」に書き替える。ステップ433では、このエントリがディスクリプタリングの最後に登録されていたものか否かをテストする。ディスクリプタリング402の最後でなければ、ステップ434でRP400をインクリメントし、次のデキュー位置を知らせて、ステップ435で取り出したディスクリプタエントリを呼び出し、元に渡す。ディスクリプタリングの最後であれば、ステップ436でRP400の値をディスクリプタリングのベースアドレスにセットすることにより、ディスクリプタをサイクリックに使用する。ステップ430で実行権フィールド=「ENQ」と表示されていれば、これはディスクリプタリングで前回のサイクルエントリを削除してから、エンキュー側により未だ新しくエントリが登録されていないことを示しているの、何もせず、リターンする。

【0046】図29は、IOC配信処理プログラムのフローチャートである。IOC配信プログラムは、定期的に自PM宛の受信処理FIFO通信エリア(図5～図7参照)を読み出し、他のPMから受信処理要求が送られてきたか否かをチェックする。図30は、INC送信処理プログラムのフローチャートである。また、図31および図32は、INC受信処理プログラムのフローチャートである。以下、図18～図33を参照しながら、実施例の動作を説明する。図10および図11に示した分類に従って、次の4つのメッセージ通信のケースについて詳述する。

(ケース1) システム内PM内通信

(ケース2) システム内PM間通信

(ケース3) システム間中継PMなしの通信(システム間PM無中継通信)

(ケース4) システム間中継PMありの通信(システム間PM中継通信)

【0047】(ケース1) システム内PM内通信

(a) 送信側の処理

プロセッサモジュール(PM)18-1内の送信オブジェクトから同じPM(18-1)内の受信オブジェクトにメッセージを転送する場合を考える。送信オブジェクトと受信オブジェクトは互いに非同期で実行されるので、その実行順序は種々のケースが考えられるが、ここでは簡単のために送信オブジェクトが受信オブジェクトよりも先に実行される図18のケースを仮定する。図1

30

8において、S100はAPLオブジェクト起動、S101はMB捕捉、S102はメッセージ書き込み、S103は送信処理、S104はAPLオブジェクト終了である。また、S110はAPLオブジェクト起動、S111は受信処理、S112はメッセージ読出し、S113はMB解放、S114はAPLオブジェクト終了である。

「ステップS100」カーネルは、ステップS100で、アプリケーション(APL)オブジェクトを実行させる準備を行う。APLオブジェクトとは、ユーザモードで動作するアプリケーションプログラムのことである。ステップS100の「APLオブジェクト起動」の内容は、図22に示すように、ステップ330でレディキュー(実行可能なオブジェクトコントロールブロック(OCB)(図16の210)がリスト構造でリンクされたもの(図16参照)から、次に実行すべきAPLオブジェクトのOCB210を取り出す。次に、ステップ331で、APLオブジェクトに与えられているオブジェクトID(図12の170)を分散メモリプロテクタ23内のカレントコントロールレジスタ(CCR)51に設定する。このタイミングは、図18のタイミングt100により示される。次に、図22のステップ332で、必要な情報をOCB210からプロセッサ19にロードして、APLオブジェクトの実行を開始する。ここでは、APLオブジェクトが送信オブジェクトであるため、送信オブジェクトの実行が開始される。

【0048】「ステップS101」送信オブジェクトが、MBの捕捉をカーネルに要求すると、カーネルは図23のMB捕捉処理プログラムを実行する。図23のステップ350で、メッセージを受信する受信オブジェクトのID(図12のRID170)内のシステムIDフィールド171を取り出す。この値をシステムルーチングテーブル(図17の249)のシステムIDフィールド250の各エントリと比較し、一致したエントリのシステム通信モードフィールド251、通信リンクフィールド252、送信中継PMフィールド253の値を取り出す。ここでは、同一システム内の同一PM内通信であるため、「自システムID」と記されたエントリ260と一致し、そのシステム通信モードフィールド251は、「システム内通信」261と表示されているため、ステップ351からステップ352に進む。ステップ353では、受信オブジェクトID(RID)のPMIDフィールド172をPMルーチングテーブル(図17の289)のPMIDフィールド290の各エントリと比較して、一致したエントリのPM通信モードフィールド291と、MB管理マップアドレスフィールド292の値を取り出す。ここでは、同一システム内の同一PM内通信であるため、「自PMID」と記述されたエントリ300と一致し、その結果、「PM内通信301」と表示されたPM通信モードフィールドと、「PM

31

内通信MB管理マップベースアドレス302』と表示されたMB管理マップアドレスフィールドとを取り出す。

【0049】ステップ353では、選択したMB管理マップアドレス292で指定されたMB管理マップを検索し、空きを表示しているMB管理マップエントリを検出し、それに対応する空きMBアドレスMBA（図15の202）を求め、そのMB管理マップエントリ（図15の190）を「使用中」に書き替える。MB管理マップエントリとMBとは、1対1に対応するように構成されており、MB管理マップエントリアドレスからMBアドレスを求めることができるとともに、逆にMBアドレスからもMB管理マップエントリアドレスを求めることができる。ここでは、PM内通信用MB管理マップが選択され、その中の適当な空きMB、例えば図5、図7のMB（X11-2）が選択される。ステップ353の最後で、MB（X11-2）に対応するMB管理マップエントリ（XM11-2）が図17の及楽石械胤t101で「使用中」に書き替えられる。続いて、ステップ354で、MB（X11-2）に対応するメモリケーバビリティレジスタ（MCR）53のアドレスを求め、MCR53に、このMBへのアクセス権を獲得したオブジェクト、つまり送信オブジェクトのID（SID）が図18のタイミングt102で登録される。以上が、図18のステップS101のMB捕捉処理である。

【0050】「ステップS102」送信オブジェクトは、図18のステップS102で、作成したメッセージをMB（X11-2）にタイミングt103で書き込む。図18のプロテクションウィンドウ開放期間w1は、CCR51とMCR53の両方に値が設定されている区間で、MCRとCCRとが指定しているAPLオブジェクト（送信オブジェクト）は、MB（X11-2）にアクセスすることができるが、他のAPLオブジェクトはアクセスすることができなくなり、MBに対する強固な保護が実現される。

【0051】「ステップS103」送信オブジェクトがステップS103でカーネルに送信処理を依頼すると、カーネルは、図25の送信処理プログラムを実行する。まず、図25のステップ380でMB（X11-2）に対応するMCR53を図18のタイミングt104でクリアする。これで、図18のプロテクションウィンドウ開放期間w1が終了するため、以後、カーネル以外はMB（X11-2）にアクセスできなくなる。次に、図25のステップ381で、通信モードをチェックする。ここでは、「システム内PM内通信」であるため、ステップ382で受信オブジェクトのOCB210のメッセージキューにMB（X11-2）のアドレス（MBA）を登録する。続いて、ステップ383でOCB210のSTATUSフィールド（図15の214）が「送信側実行待ち」となっているか否かをテストする。「送信側実行待ち」は、受信オブジェクトが送信オブジェクトより

32

も先に実行された結果、受信オブジェクトが送信オブジェクトの実行を待っている状態を表わす。このケースでは、送信オブジェクトが受信オブジェクトよりも先に実行されるため、ステップ383は「NO」となって、ステップ385でOCB210のSTATUSフィールド214を「受信側実行待ち」と設定して、受信オブジェクトの実行を待つ状態になったことを表示する。

【0052】「ステップS104」ステップS104では、カーネルがAPLオブジェクト終了処理を起動する。APLオブジェクト終了処理プログラムは、図22に示されており、ステップ340でカレントケーバビリティレジスタCCR51の値を図18のタイミングt105でクリアする。続いて、図22のステップ341で送信オブジェクトの再開に必要な情報をOCB210に退避することにより、送信オブジェクトの処理を完了する。

【0053】（b）受信側の処理

「ステップS110」図18のステップS110において、カーネルは次に実行すべきオブジェクトとして受信オブジェクトを選択し、図22のAPLオブジェクト起動プログラムを実行する。送信オブジェクトの場合と同じように、図18のタイミングt110で受信オブジェクトID（RID）（図15の183）をカレントケーバビリティレジスタ（CCR）51に設定する。

【0054】「ステップS111」受信オブジェクトの実行が開始され、受信オブジェクトがカーネルに受信処理を要求すると、カーネルは図26の受信処理プログラムを起動する。図26のステップ390では、受信オブジェクトのOCB210のSTATUSフィールド214が「受信側実行待ち」または「同期完了」となっているか否かをテストする。「受信側実行待ち」は、送信オブジェクトが受信オブジェクトよりも先に実行された結果、受信オブジェクトの実行を待っている状態である。また、「同期完了」は、受信オブジェクトが送信オブジェクトよりも先に実行されたため、途中で休止し、その後、送信オブジェクトが実行されたことにより、送受信間の同期がとれた状態を示している。このケースでは、図18のステップS103で、STATUSフィールド214を「受信側実行待ち」と設定しているため、ステップ390は「YES」となる。これは、メッセージが送信側から既に届いているため、直ちにメッセージキューにつながっているメッセージをキューから外し、そのMB（X11-2）のアドレスを取り出す。続いて、ステップ392では、このMBアドレスに対応するMCR53に受信オブジェクトID（RID）183を設定する。このタイミングは、図18のタイミングt111で示される。これにより、CCR51とMCR53の両方に受信オブジェクトID（RID）183が設定されたので、以後、図18のプロテクションウィンドウ開放期間（w2）中は、MB（X11-2）が受信オブジェク

33

ト以外のAPLオブジェクトからアクセスできないようにプロテクトされることになる。

【0055】「ステップS112」受信オブジェクトが、MB(X11-2)からタイミングt112でメッセージを読み出して、対応する処理を行う。

【0056】「ステップS113」受信オブジェクトは、ステップS113でカーネルにMB解放を依頼する。カーネルは、図24のMB解放処理プログラムを起動する。図24において、ステップ360で、MB(X11-2)に対応するMCR53をクリアする。このタイ

ミングは、図18のタイミングt113に示されている。続いて、図24のステップ361でMB(X11-2)に対応するMB管理マップエントリ(XM11-2)を「空き」に更新する。このタイミングは、図18のタイミングt114に示されている。MB管理マップエントリを「空き」に更新すると、以後は、いつでもこのMB(X11-2)を再利用することが可能となる。

【0057】「ステップS114」カーネルがAPLオブジェクト終了処理を実行して、カレントケーバリティレジスタ(CCR)51の値を図18のタイミングt115でクリアする。APLオブジェクト終了処理は、図22に示されており、送信オブジェクトの場合と同じであるため説明を省略する。以上のように、システム内PM内通信の場合には、送信オブジェクトと受信オブジェクトがユーザレベルで直接アクセスできる分散共有メモリエリアに、メッセージバッファを確保することにより、メッセージバッファ間の無駄なコピーを避けることができるので、効率的なメッセージ通信が可能となる。

【0058】(ケース2) システム内PM間通信

(a) 送信PM側の処理

マルチプロセッサシステム17-1内のプロセッサモジュール(PM)18-1内の送信オブジェクトから、PM18-2内の受信オブジェクトにメッセージを転送する場合を仮定する。さらに、図19に示すように、送信オブジェクトの実行終了前に受信オブジェクトの実行が開始される場合を考える。使用するMBを図5のX12-1とすると、対応するMB管理マップエントリはXM12-1となる。また、この処理依頼内容の詳細は受信ディスクリプタリングXD12-Rに、受信処理要求は受信PM18-2宛の受信処理FIFOエリアXF2-Rに登録されるものとする。処理の流れを、図19の各ステップに従って説明する。S120はAPLオブジェクト起動、S121はMB捕捉、S122はメッセージ書込み、S123は送信処理、S124はAPLオブジェクト終了である。また、S110はAPLオブジェクト起動、S111は受信処理、S112はメッセージ読出し、S113はMB解放、S114はAPLオブジェクト終了である。

【0059】「ステップS120」送信PM18-1のカーネルは、ステップS120でAPLオブジェクト起

34

動プログラム(図22)を実行し、送信オブジェクトID(SID)を図19のタイミングt120でPM18-1のCCR51-1に設定し、送信オブジェクトの実行を開始する。

【0060】「ステップS121」送信オブジェクトはMB捕捉をカーネルに要求し、カーネルは図23のMB捕捉処理プログラムを実行する。このケースでは、同一システム内で2つのPMにまたがった通信であるため、図23のステップ350で検索して取り出されるシステム通信モードフィールド(図17の251)の値は「システム内通信」261となり、ステップ351からステップ352に進む。ステップ352で検索して取り出されるPM通信モードフィールド(図17の291)の値は「PM間通信311」となり、これに対応して「PM間通信MB管理マップベースアドレス322」と表示されたMB管理マップアドレスフィールドを取り出す。ステップ353では、選択した「PM間通信MB管理マップベースアドレス322」で指定されたMB管理マップを検索し、空きを表示しているMB管理マップエントリを検出して、それに対応する空きMBアドレスを求め、そのMB管理マップエントリを「使用中」に書き替える。この例では、PM18-1からPM18-2への通信用MBX12-1が図19のタイミングt121で「使用中」に書き替えられるが、この時点で同時にXM12-1と同一アドレスを持つ受信側分散共有メモリ21-2のMB管理マップエントリYM12-1も書き替えられる。この書き替えは、次のような操作で行われる。

【0061】まず、送信PM18-1のプロセッサ19-1がMB管理マップエントリXM12-1の書き替え命令を実行すると、MB管理マップエントリXM12-1のアドレスと書き替えデータがプロセッサバス30-1(図1参照)に送出される。送信側分散メモリカップラ22-1では、送信側分散共有メモリ21-1への書き込みアクセスがあった場合に、バス信号デコーダ31-1が分散共有メモリ21-1への書き込みアクセスであることを検出し、信号線32-1を経由して転送制御部33-1を起動する。転送制御部33-1は、MB管理マップエントリXM12-1のアドレスと書き替えデータを取り込み、モジュールID管理部34-1に、MB管理マップエントリのページアドレスを供給する。モジュールID管理部34-1のCAMコマンドレジスタ(図8の67-1)には、予め「キーレジスタ(図8の66-1)との比較動作」を指示するコマンドが入れているため、MB管理マップエントリXM12-1のページアドレスがキーレジスタ66-1に格納されると、キーレジスタ(66-1)とCAMセル部(図8の61-1)との一斉比較が行われる。CAMセル部61-1の各エントリは、他のPMの分散共有メモリエリアと重複しているエリアのページアドレス63を含んでい

35

る。

【0062】本実施例においては、受信PM18-2との間で共有されているXM12-1のページアドレスと、受信PM18-2 IDのペアがそれぞれCAMセル部61-1とデータメモリ部62-2に登録されている。従って、CAMセル部61-1における比較動作により一致が検出され、対応する受信PM18-2のIDがデータメモリ部62-1から取り出される。これを、パケット送信レジスタ35-1に転送する。パケット送信レジスタ35-1では、図9に示すように、属性識別フィールド80を‘0’に設定して、共有分散メモリアクセスタイプのパケットであることを表示し、宛先PM IDフィールド81には、モジュールID管理部から取り出した値(PM18-2のID)を設定する。また、分散共有メモリアドレスフィールド82と書き込みデータフィールド83には、プロセッサバス30-1上のアドレス(XM12-1のアドレス)とデータ(『使用中』表示データ)を載せ、書き込み単位フィールド84には、プロセッサバス30-1が指示する信号(バイト書き込みとバイト位置情報)を載せて送信パケットを編集する。これをパケット送信バッファ36-1に送出すると、パケット送信バッファ36-1は送信パケットをプロセッサモジュール間通信路25に送出する。プロセッサモジュール間通信路25は、送信パケットの宛先PM IDフィールド81を見てルーチングを行い、受信PM18-2のパケット受信バッファ37-2に送り届ける。

【0063】受信PM18-2のパケット受信バッファ37-2のデータはパケット受信レジスタ38-2に入れられてパケットデコーダ39-2でデコードされ、共有分散メモリ21-2への書き込みであることがわかると、転送制御部33-2を起動する。転送制御部33-2は、パケット受信レジスタ38-2内の分散共有メモリアドレス(XM12-1のアドレス)、書き込みデータ(『使用中』表示データ)、書き込み単位の情報をプロセッサバス30-2経由で分散共有メモリ21-2に送ることにより、XM12-1への書き込みデータと同じ値が受信側分散共有メモリ21-2の同じアドレス位置、つまりYM12-1に書き込まれる。このようにして、分散共有メモリへの書き込み側が自分の分散共有メモリへのローカルな書き込みを行うだけで、分散メモリカップラ22の働きにより自動的に宛先の分散共有メモリにも同じ値がコピーされるので、DMAのようなソフトウェアによるコピーオーバーヘッドを伴わないという利点がある。

【0064】なお、コピーバック型のキャッシュメモリを備えるプロセッサ19では、分散共有メモリ21への書き込みアクセスがキャッシュヒットすると、その時点でキャッシュメモリへの書き込みだけが行われ、分散共有メモリ21への書き込みは直ちに行われないことがあ

36

る。その場合には、キャッシュメモリへの書き込み後、キャッシュメモリ内の書き込みデータを分散共有メモリに戻す命令を実行することにより、送信側分散共有メモリ21-1への書き込みを確実に行うので、受信側分散共有メモリ21-2へのコピーも確実に行うことができる。次に、図23のステップ354では、MB(X12-1)に対応するメモリーキーバビリティレジスタ53-1のアドレスを求め、MCR53-1に対しこのMBへのアクセス権を獲得した送信オブジェクトのID(SID)を図19のタイミングt122で登録する。以上で、ステップS121のMB捕捉処理を終了する。

【0065】「ステップS122」送信オブジェクトが作成したメッセージを送信側分散共有メモリのMB(X12-1)にタイミングt123で書き込みを行う度毎に、この書き込みデータが分散メモリカップラ22により受信側分散共有メモリの同一アドレスのMB(Y12-1)にも伝搬して書き込まれる。その操作は、図19のタイミングt121で説明した通りである。なお、本実施例のメッセージ書き込みステップでは、図9に示すように、書き込み単位、例えば4バイトの書き込みデータ毎に1つのパケットを構成し、分散メモリカップラ22間で送受する構成であるがメッセージのようにメモリの連続エリアへの書き込みが行われる場合には、図9の送受信パケットを、先頭の書き込みアドレスに続いて複数の書き込みデータを1パケットにまとめた構成にして、分散メモリカップラ22間で送受する方法も可能である。その場合には、連続エリアへの書き込みが行われているのか、または単発エリアの書き込みが行われているのかを分散メモリカップラ22側で事前に検知することができないため、分散メモリカップラ22は、予め決められた時間内に決められた回数の連続アドレスエリア書き込みがあれば、それらを1パケットにまとめて送る。また、決められた時間内に決められた回数より少ない連続アドレスエリア書き込みであれば、タイムアウトになった時点で、それまでにたまっている連続アドレスエリア書き込みデータを1パケットにまとめて送出すればよい。図19において、連続番地CCR51-1とMCR53-1の両方の値が設定されている区間(ステップS121からステップS124までの区間)では、MCRとCCRにより指定された送信オブジェクトがMB(X12-1)にアクセスすることができるが、PM18-1内の他のAPLオブジェクトはアクセスすることができない。

【0066】「ステップS123」送信オブジェクトは、カーネルに送信処理を依頼する。送信処理プログラムは、図25に示されるように、先ずステップ380でMB(X12-1)に対応するMCR53-1をクリアする。この動作は、図19のタイミングt124に示されており、これによりプロテクションウィンドウ開放期間が終了するので、これ以後、カーネル以外はMB(X

37

12-1)にアクセスすることができなくなる。次に、図25のステップ381で通信モードをチェックする。ここでは、「システム内PM間通信」であるため、ステップ386に進む。ステップ386では、図19のステップS121で検索一致したPM ID (図17の290)、つまりPM18-2宛の受信ディスクリプタリングXD12-Rに受信処理制御情報を登録(エンキュー)すると同時に、図14のフォーマットの処理要求を受信PM18-2宛の受信処理FIFO(XF2-R)に書き込む。これらの書き込みは、図19のタイミングt125で行われる。受信ディスクリプタリングXD12-Rへのエンキュー処理は、図27に示されているので、説明は省略する。受信ディスクリプタリングXD12-Rに書き込みが行われると、書き込みデータが分散メモリカップラ22-1により受信PM18-2にも転送され、その分散共有メモリ21-2の同一アドレスであるYD12-Rにも書き込まれる。

【0067】一方、受信処理FIFOへの書き込みについて、送信PM18-1側の受信処理FIFOエリア(XF2-R)は、通常のRAMで構成されているが、受信PM18-2側の受信処理FIFOエリア(YF2-R)はFIFOメモリで構成されている。従って、XF2-Rへの書き込みデータが分散メモリカップラ22により受信側の分散共有メモリの受信処理FIFOエリア(YF2-R)に伝達されると、YF2-Rでは、以前に書き込まれたデータを保存したまま、新しい書き込みデータが追加して書き込まれる。以上により、メッセージ送信に必要な情報が受信PM18-2に全て伝達された。

【0068】「ステップS124」カーネルが図22のAPLオブジェクト終了処理を起動し、カレントケーバビリティレジスタ(CCR)51-1の値を図18のタイミングt126でクリアする。

【0069】(b)受信PM18-2側の処理
「ステップS130」図19のステップS130において、カーネルは次に実行すべきオブジェクトとして受信オブジェクトを選択し、図19のタイミングt130で受信オブジェクトID(RID)を受信PM18-2内のカレントケーバビリティレジスタ(CCR)51-2に設定する。

【0070】「ステップS131」受信オブジェクトの実行が開始され、受信オブジェクトがカーネルに受信処理を要求すると、カーネルは図26の受信処理プログラムを起動する。図26のステップ390では、受信オブジェクトのOCB210のSTATUSフィールド(図15(d)の214)が「受信側実行待ち」または「同期完了」になっているか否かをテストする。このケースでは、この時点で送信オブジェクトからのメッセージが到着していないため、「受信側実行待ち」または「同期完了」の状態になく、「no」となる。そこで、ステッ

38

プ393で先に受信オブジェクトの要求が到着したことを示すために、STATUSフィールド214を「送信側実行待ち」に設定し、カーネルにリターンする。

【0071】「ステップS132」カーネルは、受信オブジェクトが「送信側実行待ち」になったことを知り、送信オブジェクトからメッセージが届くまで受信オブジェクトを休止させるために、図22のAPLオブジェクト終了処理を起動し、ステップ340でカレントケーバビリティレジスタ(CCR)51-2の値を図19のタイミングt131でクリアし、ステップ341で受信オブジェクト再開に必要な情報をOCB210に退避する。

【0072】「ステップS140」受信PM18-2のIOC配信プログラムは、図29に示すように、定期的にPM#2宛の受信処理FIFO(YF2-R(図7))を読み出すことにより受信処理要求の有無をチェックする。具体的には、図29のステップ500で受信処理FIFO(YF2-R)から要求を図19のタイミングt140で読み出し、ステップ501で要求の有無をチェックする。もし、要求があれば、ステップ502で、引き続き同じタイミングt140で読み出した要求が指定している受信ディスクリプタリングYD12-Rに対して図28の「デキュー処理」を実行し、受信ディスクリプタ(図15(c)の200)を取り出す。ステップ503で、受信ディスクリプタ内のMBA(図15(c)の202)を参照して、図15(a)で示される構造のMB(Y12-1)をアクセスしてMB(Y12-1)内の受信オブジェクトID(RID)(図15の183)を取り出す。ステップ504で、RID183に対応する図15(d)のOCBのメッセージキューに、MB(Y12-1)のアドレスMBA202を登録する。ステップ505では、この受信オブジェクトのOCB210のSTATUSフィールド(図15の214)が「送信側実行待ち」となっているか否かをテストする。ここでは、図19のステップS131でSTATUSフィールド214を「送信側実行待ち」と設定しているため、ステップ505は「yes」となる。これは、受信オブジェクトが送信側からメッセージを受け取る準備をして休止状態にいることを表わしているため、ステップ506では、受信オブジェクトのOCB210のSTATUSフィールド214を「同期完了」と設定して送信要求と受信要求の両方が揃ったことを表示する。次に、ステップ507で、受信オブジェクトのOCB210をレディキューに登録して受信オブジェクトが再起動されるようにする。

【0073】「ステップS150」受信PM18-2のカーネルは、レディキューの中から次に実行すべきオブジェクトとして受信オブジェクトを選択し、図19のタイミングt150で受信オブジェクトID(RID)を受信PM18-2内のカレントケーバビリティレジスタ

39

(CCR) 51-2に設定する。

【0074】「ステップS151」受信オブジェクトの実行が開始され、受信オブジェクトがカーネルに受信処理を要求すると、カーネルは図26の受信処理プログラムを起動する。図26のステップ390では、受信オブジェクトのOCB210のSTATUSフィールド214が「受信側実行待ち」または「同期完了」となっているか否かをテストする。このケースでは、図19のステップS140でSTATUSフィールド214を「同期完了」と設定しているため、ステップ390は「yes」となる。これは、メッセージが送信側から既に届いているので、直ちにメッセージを読み出してもよいことを示している。そこで、ステップ391では、OCB210のメッセージキューにつながっているメッセージをキューから外し、そのMB(Y12-1)のアドレスを取り出す。続いて、ステップ392で、このMBアドレスに対応するMCR53-2に受信オブジェクトID(RID)を設定する。このタイミングは、図19のタイミングt151で示されている。これにより、CCR51-2とMCR53-2の両方に受信オブジェクトID(RID)が設定されたので、以後は、MB(Y12-1)が受信オブジェクト以外のAPLオブジェクトからアクセスできないようにプロテクトされた。

【0075】「ステップS152」受信オブジェクトがMB(Y12-1)からタイミングt152でメッセージを読み出し、対応する処理を行う。ここで受信オブジェクトは、送信オブジェクトがメッセージを書き込んだ送信側分散共有メモリ21-1のMB(X12-1)からでなく、そのコピーを記憶する受信側分散共有メモリ21-2のMB(Y12-1)から読み出すため、リモートメモリアccessを行う場合に比較して極めて高速である。

【0076】「ステップS153」受信オブジェクトは、カーネルにMB解放を依頼する。カーネルは、図24のMB解放処理プログラムを起動し、図24のステップ360で、MB(Y12-1)に対応するMCR53-2を図19のタイミングt153でクリアする。次に、図24のステップ361でMB(Y12-1)に対応するMB管理マップエントリ(YM12-1)を図19のタイミングt154で「空き」に書き替える。この値の書き替えは、受信側の分散共有メモリ21-2のみならず、同時に送信側の分散共有メモリ21-1の同一ロケーションXM12-1に対しても行われる。その理由は、受信側分散メモリカップラ22-2内のモジュールIDディレクトリ(図8の60)に、各MB管理マップのページアドレスと、送信側PMIDのペアが記憶されているので、受信側分散共有メモリ21-2への書き込みが生じると、分散メモリカップラ22-2、22-1経由で、送信の分散共有メモリ21-1にも書き込まれるためである。本発明では、MBは常に送信PMで一

40

元管理するので、図19のタイミングt155で、送信側のMB管理マップエントリXM12-1への書き替えが完了すると、この時点以後は、いつでも送信PMでMB(X12-1)を再利用することが可能である。

【0077】「ステップS154」カーネルがAPLオブジェクト終了処理を実行し、カレントカーバリティレジスタ(CCR)51-2の値を図19のタイミングt156でクリアする。以上のように、システム内PM間通信の場合には、分散メモリカップラ22の働きにより、メッセージおよびその制御情報を送信PMと受信PM間で極めて簡単に授受を行うことができるので、効率的なメッセージ通信が行える。

【0078】(ケース3) システム間PM無中継通信

(a) 送信システム17-1のPM18-1内処理
送信側マルチプロセッサシステムと受信側マルチプロセッサシステムのいずれも、中継PMを介さないで通信するパターンを例を説明する。図20に示すように、マルチプロセッサシステム17-1において送信オブジェクトの存在するプロセッサモジュール(PM)18-1からネットワークを介して他のマルチプロセッサシステム17-2の受信オブジェクトの存在するPM18-3にメッセージを転送する場合を考える。このケースでは、送信オブジェクトの実行終了前に、受信オブジェクトが実行開始するものとする。送信側が使用するMBは、図7のX11-1、そのMB管理マップエントリはXM11-1とする。また、この処理依頼内容の詳細は、送信ディスクリプタリングXD11-Sに、送信処理要求は送信PM18-1の送信処理FIFO(XF1-S)に登録されるものとする。処理の流れを図20の各ステップに沿って説明する。図20におけるS160はAPLオブジェクト起動、S161はMB捕捉、S162はメッセージ書き込み、S163は送信処理、S164はAPLオブジェクト終了、S170はINC送信処理、S180はAPLオブジェクト起動、S181は受信処理、S182はAPLオブジェクト終了、S190はINC受信処理、S200はAPLオブジェクト起動、S201は受信処理、S202はメッセージ読出し、S203はMB解放、S204はAPLオブジェクト終了である。

【0079】「ステップS160」送信PM18-1のカーネルは、ステップS160のAPLオブジェクト起動の実行により、送信オブジェクトのID(SID)をタイミングt160でPM18-1のCCR51-1に設定し、送信オブジェクトの実行を開始する。

【0080】「ステップS161」送信オブジェクトはMBの捕捉をカーネルに要求すると、カーネルは図23のMB捕捉処理プログラムを実行する。このケースは、異なるシステムに存在する2つのPM間の通信であるため、図23のステップ350で検索して取り出されるシステム通信モードフィールド(図17の251)の値は

41

「システム間通信」271となる。従って、図23のステップ351では、「システム間通信」側選択され、ステップ355で対応するエントリの通信リンクID(図17の272)と送信中継PM ID273の値が取り出される。ステップ356で、送信中継PM ID273をキーとしてPMルーチングテーブル289を検索し、ステップ353で送信PM18-1と送信中継PMとの通信で使用するMBプールとそのディスクリプタリングを決定する。このケースの送信側システムでは、PM無中継であるため送信PM18-1と送信中継PMとが一致し、ステップ356で、PM内送信用MBプールX11とそのMB管理マップXM11が選択される。ステップ353では、PM内通信用MBマップ端漆歳XM11-1が「使用中」に書き替えられる。次に、図23のステップ354で、MB(X11-1)に対応するメモリカーパビリティレジスタ(MCR)53-1に、このMBへのアクセス権を獲得した送信オブジェクトのID(SID)が図20のタイミングt162で登録される。

【0081】「ステップS162」送信オブジェクトは、作成したメッセージを送信側分散共有メモリ(MB(X11-1))にタイミングt163で書き込む。このメッセージの書き込みステップでは、ccr51-1とMCR53-1の両方の値が設定されている区間は、MCRとCCRにより指定されている送信オブジェクトがMB(X11-1)にアクセスすることができるが、他のAPLオブジェクトはアクセスすることができない。

【0082】「ステップS163」送信オブジェクトでは、カーネルに送信処理を依頼する。図25の送信処理プログラムではステップ380でMB(X11-1)に対応するMCR53-1をクリアする。これは、図20のタイミングt164で示されており、これによりプロテクションウィンドウ開放期間が終了するため、以後、カーネル以外はMB(X11-1)にアクセスできなくなる。次に、図25のステップ381で、通信モードをチェックする。このケースでは、「システム間PM無中継通信(システム間PM内通信)」であるため、ステップ387に進み、図20のステップS161で検索一致したPM ID(図17の290)、つまりPM18-1宛の送信ディスクリプタリングXD11-Sに送信制御情報を登録(エンキュー)し、図14のフォーマットの処理要求を自PM18-1宛の送信処理FIFO(XF1-S)(図7参照)に書き込む。これらの登録は、図20のタイミングt165で行われる。なお、このケースでは、送信PM18-1の送信ディスクリプタリングXD11-Sと送信処理FIFO(XF1-S)は、他の分散共有メモリ内にコピーを持たないため、分散メモリカップラによる他分散共有メモリへの書き込みは行われない。

【0083】「ステップS164」カーネルがAPLオ

42

ブジェクト終了処理を起動し、カレントカーパビリティレジスタ(CCR)51-1の値を、図20のタイミングt166でクリアする。

【0084】「ステップS170」送信PM18-1のINC送信処理プログラムは、図30に示すように、定期的に自PM宛の送信処理FIFO(XF1-S)を読み出すことにより要求の有無をチェックしている。具体的には、図30のステップ520で、送信処理FIFO(XF1-S)から要求を図20のタイミングt170で読み出し、ステップ521で要求の有無をチェックする。本実施例のステップS170の時点においては、既に送信処理FIFO(XF1-1)に要求が登録されているので、ステップ522で、要求が指定している送信ディスクリプタリングXF1-Sに対して図28の「デキュー処理」を実行する。ステップ523で、図13に示す構造の送信ディスクリプタを引き続き図20のタイミングt170で取り出し、この情報をもとにしてネットワークを介した通信に必要とする各種のプロトコル処理(例えば、ネットワークパケットのヘッダの作成、再送制御情報の作成、チェックサムの計算等)を行う。次に、ステップ524では、ネットワークアダプタの起動に必要なアダプタ送信ディスクリプタ(図30のステップ530)を作成する。ステップ525では、このアダプタ送信ディスクリプタをアダプタ送信ディスクリプタリングに、図20のタイミングt171で登録する。

【0085】ネットワークアダプタ24-1は、常時、アダプタ送信ディスクリプタリングを監視しており、ディスクリプタの登録があると、この情報をもとにしてネットワークが要求するサイズにメッセージを分割し、ヘッダや誤り制御情報を付加してネットワークパケットとしてネットワークに送出する。このタイミングは図20のタイミングt172で示されている。ネットワークアダプタ24-1は、メッセージの転送を完了するか、あるいは通信相手のシステムからメッセージ到着確認の連絡を受けると、送信側でメッセージバッファを保持しておく必要がなくなったため、図20のタイミングt173でMB(X11-1)に対応するMB管理マップのエントリXM11-1を「空き」に書き替える。その結果、この時点以降、送信PM18-1では、いつでもこのMB(X11-1)を再利用することができる。

【0086】(b)受信システム17-2のPM18-3内処理

「ステップS180」図20において、受信システム17-2のカーネルは、次に実行すべきオブジェクトとして受信オブジェクトを選択し、図20のタイミングt180で受信オブジェクトID(RID)を受信PM18-3内のカレントカーパビリティレジスタ(CCR)51-3に設定する。

【0087】「ステップS181」受信オブジェクトの実行が開始され、受信オブジェクトがカーネルに受信処

43

理を要求すると、カーネルは図26の受信処理プログラムを起動する。図26のステップ390では、受信オブジェクトのOCB(図15(d)の210)のSTATUSフィールド(図14の214)が「受信側実行待ち」または「同期完了」になっているか否かをテストする。このケースでは、この時点で、送信オブジェクトからのメッセージは到着していないため、「受信側実行待ち」または「同期完了」の状態ではなく、「no」となる。そこで、ステップ393で、先に受信オブジェクトの要求が到着したことを示すために、STATUSフィールド214を「送信側実行待ち」に設定し、カーネルにリターンする。

【0088】「ステップS182」カーネルは、受信オブジェクトが「送信側実行待ち」になったことを知り、送信オブジェクトからメッセージが届くまで受信オブジェクトを休止させるため、図22のAPLオブジェクト終了処理プログラムを起動し、ステップ340でカレントケーパビリティレジスタCCR51-3の値を図20のタイミングt181でクリアし、ステップ341で受信オブジェクト再開に必要な情報をOCB210に退避する。

【0089】「ステップS190」図20のタイミングt172で、送信システム17-1からネットワークパケットが届くと、受信システム17-2のネットワークアダプタ24-3は、自分が管理している、分散共有メモリ21-3内のPM#3内通信領域の空きのローカルメッセージバッファの1つに、ネットワークパケットを記憶する。メッセージが複数のネットワークパケットで分割されて送られて来る場合には、送られてきたネットワークパケットをパケットヘッダを見ながらローカルメッセージバッファ内の指定された位置に入れることによりメッセージの組み立て処理を行い、最終的に1つのメッセージに組み立てる。そして、でき上がったメッセージに関する制御情報をアダプタ受信ディスクリプタリングに登録する。INC受信処理プログラムは、図31、図32に示すように、定期的にアダプタ受信ディスクリプタリングを読み出すことにより要求の有無をチェックする。

【0090】具体的には、図31および図32において、受信システム17-2のPM18-3のINC受信処理プログラムは、ステップ560でアダプタ受信ディスクリプタリングに対して、図28の「デキュー処理」を実行し、ステップ561でアダプタ受信ディスクリプタリングにエントリが登録されていたか否かをチェックする。本実施例のステップS190の時点では、到着したメッセージをアダプタ受信ディスクリプタに登録済みのため、ステップ562でアダプタ受信ディスクリプタ(図33の590)を図20のタイミングt190で取り出し、この情報をもとにプロトコル受信処理(例えば、ネットワークパケットのヘッダのチェック、再送制

44

御情報の作成等)を行う。次に、ステップ563で、アダプタ受信ディスクリプタ中のローカルメッセージバッファアドレス(LBA)591を参照してローカルメッセージバッファをアクセスし、受信オブジェクトID(RID)(図15(a)の183)を取り出す。ステップ564では、RIDのPMIDフィールド(図12の172)をキーにして、PMルーチングテーブル(図16の289)のPMIDフィールド290を探索し、PMIDが一致したエントリの通信モードフィールド291をチェックする。

【0091】本実施例では、自PM18-3内の受信オブジェクト宛のメッセージであるため、通信モードフィールド291は「PM内通信」となって、図31のステップ565に進み、受信オブジェクトのメッセージキューにローカルメッセージバッファアドレス(LBA)591を登録する。次に、ステップ566で、受信オブジェクトのOCB210のSTATUSフィールド(図15の214)が「送信側実行待ち」となっているか否かをテストする。このケースでは、図20のステップS181で、STATUSフィールド214を「送信側実行待ち」と設定しているため、ステップ566は「yes」となる。これは、受信オブジェクトが送信側からメッセージを受け取る準備をして休止状態にあることを表わしているので、ステップ567で、受信オブジェクトのOCB210のSTATUSフィールド214を「同期完了」と設定して、送信要求と受信要求の両方が揃ったことを表示する。次に、ステップ568で受信オブジェクトのOCB210をレディキューに登録して、受信オブジェクトが再起動されるようにする。

【0092】「ステップS200」受信システム17-2の受信PM18-3のカーネルは、レディキューの中から次に実行すべきオブジェクトとして受信オブジェクトを選択し、図20のタイミングt200で受信オブジェクトID(RID)を受信PM18-3内のカレントケーパビリティレジスタ(CCR)51-3に設定する。

【0093】「ステップS201」受信オブジェクトの実行が開始され、受信オブジェクトがカーネルに受信処理を要求すると、カーネルは、図26の受信処理プログラムを起動する。図26のステップ390では、受信オブジェクトのOCB210のSTATUSフィールド(図15の214)が「受信側実行待ち」または「同期完了」となっているか否かをテストする。このケースでは、図20のステップS190で、STATUSフィールド214を「同期完了」と設定しているため、ステップ390は「yes」となる。これは、メッセージが送信側から既に届いているので、直ちにメッセージを読み出してもよいことを示している。そこで、ステップ391では、OCB210のメッセージキューにつながっているメッセージを参照して、キューから外し、そのMB

45

アドレスつまりローカルメッセージバッファアドレス (LBA) 591を取り出す。次に、ステップ392で、このLBA591に対応するMCR53-3に受信オブジェクトID (RID) を設定する。このタイミングは、図20のタイミングt201で示されている。これにより、CCR51-3とMCR53-3の両方に受信オブジェクトID (RID) が設定されたので、これ以降はローカルメッセージバッファが受信オブジェクト以外のAPLオブジェクトからアクセスできないようにプロテクトされた。

【0094】「ステップS202」受信オブジェクトがローカルメッセージバッファから図20のタイミングt202でメッセージを読み出し、対応する処理を行う。ここで、受信オブジェクトは、受信側分散共有メモリ21-3上のローカルメッセージバッファから読み出している。

【0095】「ステップS203」受信オブジェクトは、カーネルにローカルメッセージバッファ解放を依頼する。カーネルは、図24のMB解放処理プログラムを起動して、図24のステップ360でローカルメッセージバッファに対応するMCR53-3を図20のタイミングt202でクリアする。次に、図24のステップ361で、ローカルメッセージバッファに対応するMB管理マップエントリを「空き」という値に書き替える。このタイミングは、図20のタイミングt203に示されている。このMB管理マップエントリが「空き」に書き替えられた時点以降は、ネットワークアダプタ24-3は、このローカルメッセージバッファを再利用することができる。

【0096】「ステップS204」カーネルがAPLオブジェクト終了処理を実行して、カレントケーバビリティレジスタ (CCR) 51-3の値を図20のタイミングt204でクリアする。以上のように、システム間PM無中継通信の場合にも、システム内PM間通信と大差ないソフトウェアのオーバヘッドでメッセージ通信を高速に、効率よく実現することができる。

【0097】(ケース4) システム間PM中継通信
送信マルチプロセッサシステム17-1内の送信プロセッサモジュール (PM) 18-1内の送信オブジェクトが作成したメッセージを、宛先システムへの通信リンクを持つ送信マルチプロセッサシステム内の送信中継PM18-2に引き継ぎ、送信中継PM18-2がネットワークを介して他の受信マルチプロセッサシステム17-2にメッセージを転送し、受信マルチプロセッサシステム17-2の受信中継PM18-3がメッセージを受信した後、受信オブジェクトの存在する受信PM18-2にメッセージを引き継ぐ場合を仮定して、送信オブジェクトの実行終了前に、受信オブジェクトが実行開始するものとする。送信システム17-1の送信PM18-1が使用するMBは図5のX12-2、そのMB管理マッ

46

ブエントリはXM12-2、メッセージ制御情報は送信ディスクリプタリングXD12-S、送信処理要求は宛先PM18-2用送信処理FIFO (XF2-S) に、それぞれ登録されるものとする。一方、受信システム17-2の受信中継PM18-3が使用するMBを、図5～図7のZ32-1、そのMB管理マップエントリはZM32-1、メッセージ制御は受信ディスクリプタリングZD32-R、受信処理要求は宛先PM18-1情報用受信処理FIFO (ZF2-R) に、それぞれ登録されるものとする。処理の流れを図21の各ステップに沿って説明する。図21において、ステップS210はAPLオブジェクト起動、ステップ211はMB捕捉、S212はメッセージ書込み、S213は送信処理、S220はINC送信、S230はINC受信、S240はAPLオブジェクト起動、S241は受信処理、S242はAPLオブジェクト終了、S250はIOC配信、S260はAPLオブジェクト起動、S261は受信処理、S262はメッセージ読出し、S263はMB解放、S264はAPLオブジェクト終了である。

20 【0098】(a) 送信システム17-1の送信PM18-1の処理

「ステップS210」送信マルチプロセッサシステム17-1の送信PM18-1のカーネルは、図21のステップS210のAPLオブジェクト起動プログラムを実行し、送信オブジェクトのID (SID) をタイミングS210でPM18-1のCCR51-1に設定し、送信オブジェクトの実行を開始する。

【0099】「ステップS211」送信オブジェクトはMBの捕捉をカーネルに要求すると、カーネルは図23のMB捕捉処理プログラムを実行する。このケースでは、送信システム内送信PM18-1から送信システム内中継PM18-2への通信であるため、図23のステップ350で検索して取り出されるシステム通信モードフィールド (図16の251) の値は「システム間通信」271となる。従って、図23のステップ351では、「システム間通信」側が選択され、ステップ355で対応するエントリの通信リンクID (図17の272) と送信中継PM ID273の値が取り出される。このケースでは、送信中継PM ID273=「PM18-2」であるため、ステップ355、356では、
30 「PM18-2」の値をキーとしてPMルーチングテーブル289の検索が行われる。その結果、PM IDフィールド290=「PM18-2」となっているエントリが一致し、このエントリに対応するMB管理マップアドレス=「PM18-1/PM18-2間通信MB (X12) 管理マップベースアドレス312」が、PMルーチングテーブル289から取り出される。

【0100】ステップ353で、PM内通信用MB (X12) のうちの空きMB (X12-2) が選択され、そのMB管理マップエントリXM12-2を「使用中」に

47

書き替える。その書き替えは、図 21 のタイミング t 211 で、送信側分散共有メモリ 21-1 上の MB 管理マップエントリ (XM12-2) およびそれと同一アドレスを持つ、送信中継側分散共有メモリ 21-2 の MB 管理マップエントリ (YM12-2) の両方に対して行われる。その具体的方法は、図 19 のステップ S121 で説明したので、ここでは省略する。次に、図 23 のステップ 354 で、MB (X12-2) に対応するメモリカーパビリティレジスタ (MCR) 53-1 に、この MB へのアクセス権を獲得した送信オブジェクトの ID (S1D) を図 21 のタイミング t 212 で登録する。

【0101】「ステップ S212」送信オブジェクトは、作成したメッセージを送信側分散共有メモリの MB (X12-2) にタイミング t 213 で書き込むと、この書き込みデータが分散メモリカップラ 22-1 により送信中継側分散共有メモリ 21-2 の同一アドレスの MB (Y12-2) にも伝搬して書き込まれる。その操作は、図 19 の t 121 で説明したので、省略する。なお、このメッセージ書き込みステップでは、CCR 51-1 と MCR 53-1 の両方の値が設定されている期間中、MCR と CCR が指定している送信オブジェクトは MB (X12-2) にアクセスすることができるが、他の APL オブジェクトはアクセスすることができない。また、送信中継 PM18-2 にも、各 MB に対して MCR 53-2 が用意されているが、MCR 53-2 には値が設定されていないため、もし、送信中継 PM18-2 内のアプリケーションオブジェクトから MB (Y12-2) へのアクセスがあった場合にも、MCR 53-2 と CCR 51-2 の不一致が生じて送信中継 PM18-2 内での不正アクセスを検出することができる。

【0102】「ステップ S213」送信オブジェクトは、カーネルに送信処理を依頼する。送信処理プログラムは、図 25 に示すように、先ずステップ 380 で MB (X12-2) に対応する MCR 53-1 を図 21 のタイミング t 214 でクリアする。これにより、プロテクションウィンドウ開放期間が終了するので、以降はカーネル以外は、MB (X12-2) にアクセスできなくなる。次に、図 25 のステップ 381 で、通信モードが「システム間 PM 中継送信」であることを判定し、ステップ 387 では、図 21 のステップ S211 で検索が一致した PM ID (図 17 の 290)、つまり PM18-2 宛の送信ディスクリプタリング XD12-S に、送信制御情報を登録 (エンキュー) し、図 14 のフォーマットの処理要求を PM18-2 宛の送信処理 FIFO (XF2-S) (図 7 参照) に書き込む。この書き込み動作は、図 21 のタイミング t 215 で示されている。その結果、分散メモリカップラ 22-1 により送信ディスクリプタリング (XD12-S) への書き込みデータが送信中継側の分散共有メモリ 21-1 上の同一アドレスロケーション YD-S にコピーされる。同じく、送信

48

処理 FIFO (XF2-S) の書き込みデータが、送信中継 PM 宛送信 FIFO にもコピーされる。これにより、メッセージ送信に必要な全ての情報が受信中継 PM18-2 に伝達される。

【0103】「ステップ S214」カーネルが APL オブジェクト終了処理を起動し、カレントカーパビリティレジスタ (CCR) 51-1 の値を図 21 のタイミング t 216 でクリアする。

【0104】(b) 送信システム 17-1 の送信中継 PM18-2 側の処理

「ステップ S220」送信中継 PM18-2 の INC 送信処理プログラムは、図 30 に示すように、定期的に自 PM 宛の送信処理 FIFO (YF2-S) を読み出すことにより要求の有無をチェックしている。具体的には、図 30 のステップ 520 で、送信処理 FIFO (YF2-S) から要求を図 21 のタイミング t 220 で読み出し、ステップ 521 で要求の有無をチェックする。もし、要求があれば、ステップ 522 で読み出した要求が指定している送信ディスクリプタリングに対して図 28 の「デキュー処理」を実行する。本実施例では、ステップ S213 で既に送信処理 FIFO (XF2-S) に要求が登録されているので、ステップ 523 で、図 15

(c) に示す構造の送信ディスクリプタエントリを図 21 のタイミング t 220 で取り出し、この情報をもとにネットワークを介して通信に必要な各種のプロトコル処理 (例えば、ネットワークパケットのヘッダの作成、再送制御情報の作成、チェックサムの計算等) を行う。次に、ステップ 524 では、ネットワークアダプタの起動に必要なアダプタ用送信ディスクリプタを作成する。ステップ 525 では、このアダプタ送信ディスクリプタをアダプタ送信ディスクリプタリングに図 20 のタイミング t 221 で登録する。

【0105】ネットワークアダプタ 24-2 は、常時、アダプタ用送信ディスクリプタリングを監視しており、ディスクリプタの登録があると、この情報をもとにしてネットワークが要求するサイズにメッセージを分割し、ヘッダや誤り制御情報を付加してネットワークパケットとしてネットワークに送出する。このタイミングは、図 21 のタイミング t 222 で示されている。ネットワークアダプタ 24-2 はメッセージの転送を完了するか、または通信相手のシステムからメッセージ到着確認の連絡を受けると、送信側でメッセージバッファを保持しておく必要がなくなったので、図 21 のタイミング t 223 で MB 管理マップエントリ YM12-2 を「空き」に書き替えると、送信側分散共有メモリにも伝搬して MB 管理マップエントリ YM12-2 がタイミング t 224 で書き替えられる。その結果、これ以降は、送信 PM18-1 は、対応する MB (X12-2) を再利用することができる。なお、送信中継 PM18-2 では、送信 PM18-1 のように、MCR 53、CCR 51 にカーパビ

49

リティを設定していないが、もし送信中継PM18-2内のAPLオブジェクトからMB(Y12-2)に不正アクセスが行われた場合、MB(Y12-2)対応のMCR53-2に値が未設定であることにより、不正アクセスが検出できる。

【0106】(c)受信システム17-2の受信中継PM18-3側の処理

「ステップS230」送信システム17-1からネットワークパケットが届くと、受信システム17-2の受信中継PM18-3のネットワークアダプタ24-3は、
10 予め自分が管理している、分散共有メモリ21-3上のPM#3内通信領域内の空きのローカルメッセージバッファの1つに、ネットワークパケットを記憶する。メッセージが複数のネットワークパケットで分割されて送られて来る場合には、ネットワークアダプタ24-3は、受信したパケットをパケットヘッダを見ながらローカルメッセージバッファ内の指定された位置に入れることによりメッセージの組み立て処理を行って、最終的に1つのメッセージに組み立てる。そして、完成したメッセージに関する制御情報をアダプタ受信ディスクリプタリングに登録する。受信中継PM18-3のINC受信処理プログラムは、図31、図32に示すように、定期的にアダプタ受信ディスクリプタリングを読み出すことにより要求の有無をチェックする。

【0107】具体的には、図31において、INC受信処理プログラムのステップ560で、アダプタ受信ディスクリプタリングに対して図28の「デキュー処理」を実行し、ステップ561でアダプタ受信ディスクリプタリングにエントリが登録されていたか否かをチェックする。図21のステップ230の時点においては、到着したメッセージをアダプタ受信ディスクリプタに登録済みであるため、ステップ562では、アダプタ受信ディスクリプタエントリ(図33の590)を図21のタイミ
30 ングt230で取り出し、この情報をもとにプロトコル受信処理(例えば、ネットワークパケットのヘッダのチェック、再送制御情報の作成等)を行う。次に、ステップ563では、アダプタ受信ディスクリプタ中のローカルメッセージバッファアドレスLBA591を参照することによりローカルメッセージバッファをアクセスし、受信オブジェクトID(RID)を取り出す。ステップ564では、RIDのPM IDフィールド(図12の172)をキーとして、PMルーチングテーブル289のPM IDフィールド=「受信システム17-2の受信PM18-2」となっているエントリと一致するので、通信モードフィールド291=「PM間通信」となり、ステップ570に進む。

【0108】図32のステップ570で、一致したエントリに対応するMB管理マップアドレス=「PM18-3/PM18-2間通信MB(X32)管理マップZM32のベースアドレス」をPMルーチングテーブル28

50

9から取り出す。ステップ571では、MB管理マップZM32を参照して空きMB(Z32-1)を選択し、そのMB管理マップエントリZM32-1を図21のタイミングt231で「使用中」に書き替える。ZM32-1を書き替えると、受信システムの分散メモリカップラ22-3、22-2経由で、ZM32-1と同一アドレスを持つ受信側分共有メモリ21-2のMB管理マップエントリYM32-1も「使用中」に書き替えられる。次に、ステップ572で、ローカルメッセージバッファにあるメッセージをMB(Z32-1)に図21の
10 タイミングt232でコピーする。このコピー操作を実行すると、分散メモリカップラ22-3、22-2経由でメッセージが受信PMの分散共有メモリ21-2のMB(Y32-1)にも書き込まれる。つまり、このコピー操作は、受信側分散共有メモリ21-2に書き込みデータを送るために行ったものである。

【0109】図32のステップ573、574、575では、送信中継PM18-3から受信専18-2宛の受信ディスクリプタリングZD32-Rにメッセージ制御情報を図21のタイミングt233で書き込むとともに、受信処理要求を受信PM18-2宛の受信処理FIFO(ZF2-R)に書き込む。その結果、分散メモリカップラ22-3、22-1により転送されるので、受信ディスクリプタリングZD32-Rの内容が受信PM18-2の分散共有メモリ21-2の同一アドレスロケーションであるYD32-Rに、また受信処理FIFO(ZF2-R)の内容が受信PM18-2の分散共有メモリ21-2の同一アドレスロケーションであるYF2-Rに、それぞれコピーされる。これにより、メ
30 ジ送信に必要な情報が受信PM18-3に伝達された。なお、受信中継PM18-3においては、MCR53、CCR51にカーバリティを設定していないが、もし受信中継PM18-3内のAPLオブジェクトからMB(Z23-1)に不正アクセスが行われた場合には、MB(Z32-1)対応するMCR53-3に値が未設定であることにより、不正アクセスは検出できる。

【0110】(d)受信システム17-2の受信PM18-2側の処理

「ステップS240」図21のステップS240で、カーネルは次に実行すべきオブジェクトとして受信オブジェクトを選択し、図21のタイミングt240で受信オブジェクトID(RID)を受信PM18-2内のカレントカーバリティレジスタCCR51-2に設定する。

【0111】「ステップS241」受信オブジェクトがカーネルに受信処理を要求すると、カーネルは図26の受信処理プログラムを起動する。図26のステップ390では、受信オブジェクトのOCB(図15(d)の214)が「受信側実行待ち」または「同期完了」になっているか否かをテストする。このケースでは、この時点

51

で送信オブジェクトからのメッセージが到着していないので、「受信側実行待ち」あるいは「同期完了」の状態になく、「no」となる。そこで、ステップ393で、先に受信オブジェクトの要求が到着したことを示すために、STATUSフィールド214を「送信側実行待ち」に設定し、カーネルにリターンする。

【0112】「ステップS242」カーネルは、受信オブジェクトが「送信側実行待ち」になったことを知り、送信オブジェクトからメッセージが届くまで受信オブジェクトを休止させるため、カーネルが図22のAPLオブジェクト終了処理を起動し、ステップ340でカレントケーバビリティレジスタ(CCR)51-2の値を図21のタイミングt241でクリアし、ステップ341で受信オブジェクト再開に必要な情報をOCB210に退避する。

【0113】「ステップS250」受信PM18-2のIOC配信プログラムは、図29に示すように、定期的にPM#2宛の受信処理FIFO(YF2-R)を読み出すことにより受信処理要求の有無をチェックする。具体的には、図29のステップ500で、FIFOから要求を図21のタイミングt250で読み出し、ステップ501で要求の有無をチェックする。もし、要求があれば、ステップ502で、読み出した要求が指定している受信ディスクリプタリング(YD12-R)に対して図28の「デキュー処理」を実行する。ステップ503では、図15(c)の受信ディスクリプタエントリを同じタイミングt250で取り出し、そのエントリのMBA(図15(c)の202)を参照して、図15(a)で示される構造のMB(Y32-1)にアクセスする。そして、MB(Y32-1)内の受信オブジェクトID(RID)(図15(a)の183)を取り出す。ステップ504では、RID183に対応するOCB(図15の210)のメッセージキューにMB(Y32-1)のアドレスMBA202を登録する。

【0114】図29のステップ505では、この受信オブジェクトのOCB210のSTATUSフィールド214が「送信側実行待ち」となっているか否かをテストする。このケースでは、図21のステップS241で、STATUSフィールド214を「送信側実行待ち」と設定しているので、ステップ505は「yes」となる。これは、受信オブジェクトが送信側からメッセージを受け取る準備をして休止状態にいることを表わしている。従って、ステップ506では、受信オブジェクトのOCB210のSTATUSフィールド214を「同期完了」と設定して、送信要求と受信要求の両方が揃ったことを表示する。次に、ステップ507では、受信オブジェクトのOCB210をレディキューに登録して、受信オブジェクトが再起動されるようにする。

【0115】「ステップS260」カーネルは、レディキューの中から次に実行すべきオブジェクトとして受信

52

オブジェクトを選択し、図21のタイミングt260で、受信オブジェクトID(RID)を受信PM18-2内のカレントケーバビリティレジスタ(CCR)51-2に設定する。

【0116】「ステップS261」受信オブジェクトの実行が開始され、受信オブジェクトがカーネルに受信処理を要求すると、カーネルは図26の受信処理プログラムを起動する。図26のステップ390では、受信オブジェクトのOCB210のSTATUSフィールド214が「受信側実行待ち」または「同期完了」となっているか否かをテストする。このケースでは、図21のステップS250でSTATUSフィールド214を「同期完了」と設定しているので、ステップ390は「yes」となる。これは、メッセージが送信側から既に届いているので、直ちにメッセージを読み出ししてもよいことを示している。そこで、ステップ391で、OCB210のメッセージキューにつながっているメッセージを参照して、キューから外し、そのMB(Y32-1)のアドレスを取り出す。次に、ステップ392では、このMBアドレスに対応するMCR53-2に受信オブジェクトID(RID)を設定する。このタイミングは、図21のタイミングt261で示されている。これにより、CCR51-2とMCR53-2の両方に受信オブジェクトID(RID)が設定されたので、それ以降は、MB(Y32-1)が受信オブジェクト以外のAPLオブジェクトからアクセスできないようにプロテクトされた。

【0117】「ステップS262」受信オブジェクトがMB(Y32-1)から、図21のタイミングt262で受信側分散共有メモリ21-2のMB(Y32-1)からメッセージを読み出し、対応する処理を行う。MB(Y32-1)からの読み出しは、自プロセッサモジュール内に閉じたローカルなメモリアクセスであるため、リモートメモリアクセスを行う場合に比較して極めて高速である。

【0118】「ステップS263」受信オブジェクトは、カーネルにMB解放を依頼する。カーネルは、図24のMB解放処理プログラムを起動し、図24のステップ360で、MB(Y32-1)に対応するMCR53-2をクリアする。このタイミングは、図21のタイミングt263に示されている。次に、図24のステップ361では、MB(32-1)に対応するMB管理マップエントリYM32-1を「空き」という値に書き替える。このタイミングは、図21のタイミングt264に示されている。この値の書き替えは、受信側の分散共有メモリ21-2のみならず、同時に分散メモリカップラ22-2、22-1経由で、受信中継側分散共有メモリ21-3の同一番地ZM32-1に対しても行われる。図21のタイミングt265で、受信中継PM18-3のMB管理マップエントリZM32-1が「空き」に書

53

き替えられた時点以降は、受信中継PM18-3のMB(Z32-1)をいつでも再利用することができる。

【0119】「ステップS264」カーネルがAPLオブジェクト終了処理を実行して、カレントケーバビリティレジスタ(CCR)51-2の値を図21のタイミント266でクリアする。以上のように、システム間PM中継通信においても、分散メモリカップラ22の働きにより、送受信中継PMを介してもメッセージおよびその制御情報を送信PMと受信PM間で極めて簡単、かつ高速に転送することができるので、効率的なメッセージ通信が実現される。本実施例においては、図29のI/O配信処理、図30のINC送信処理、図31、図32のINC受信処理は、分散共有メモリの特番地を読み出すことにより要求を検出する、いわゆるポーリング型の処理方式で説明したが、これらを図8に示す割り込みデータタイプのパケットと図1の割り込み制御部42の機能を用いて割り込み型の処理方式で実現することも可能である。

【0120】(本発明のまとめ)このように、本発明においては、各プロセッサモジュールが分散共有メモリと分散メモリカップラを持ち、分散メモリカップラは、その分散共有メモリのアドレスを共有しているプロセッサモジュール情報を記憶する。そして、送信側のプロセッサモジュールの分散共有メモリの共有アドレスロケーションへの書き込みが発生した時点で、分散メモリカップラが、アドレスを共有する他のプロセッサモジュールに書き込み情報を転送し、書き込み情報を受けた受信側の分散メモリカップラが受信側の分散共有メモリへコピーを行う。この転送動作は、ソフトウェアの介在なしで実行されるので、従来では、分散共有メモリ間転送用としてわざわざ必要であったDMA機構制御用プログラムの実行は不要となり、ソフトウェアオーバーヘッドを大幅に削減することができる。また、DMA機構を使用する従来の方式では、転送データが全部揃ってから転送を開始するのに対して、本発明では、送信元の分散共有メモリへの書き込みが発生した時点で、相手側の分散共有メモリへの転送が開始されるので、メッセージが送信側から受信側に伝達されるまでの遅延時間(レイテンシー)も大幅に削減される。さらに、分散メモリカップラに登録するプロセッサモジュール情報に応じて、一方方向性通信、両方向通信、放送型通信等の各種の通信パターンを自由に実現することができるので、柔軟性が高い。

【0121】また、本発明における送信プロセッサモジュールと受信プロセッサモジュール間のメッセージ通信では、送信プロセッサモジュールでメッセージバッファ、メッセージの制御情報であるMBディスクリプタ、メッセージバッファ使用状態を示すMB管理マップを分散共有メモリに配置している。送信プロセッサモジュールが、これらの情報を送信側分散共有メモリに書き込み、受信側プロセッサモジュールが分散共有メモリの同

54

じアドレスロケーションから読み出すだけでよいので、プロセッサモジュール間の通信の同期を簡単かつ容易に行うことができる。また、受信側プロセッサモジュールが受信側分散共有メモリのMBディスクリプタとMB管理マップに返事を書き込み、送信側プロセッサモジュールが受信側分散共有メモリの同じアドレス位置のMBディスクリプタとMB管理マップから読み出すことにより、MBディスクリプタやMBに関する状態を簡単に送信側に伝達することができ、MBディスクリプタやMB再使用すればよいかな等を簡単に知ることができる。

【0122】さらに、本発明では、受信側プロセッサモジュールの分散共有メモリの、あるアドレスロケーションをFIFO構造にしておくことにより、送信側プロセッサモジュールが自分の分散共有メモリに書き込んだ情報が、受信側分散共有メモリのFIFOに順次書き込まれる。その結果、複数の送信側プロセッサモジュールが同時に、それぞれの分散共有メモリの同一番地に書き込みを行った場合にも、受信側の分散共有メモリのFIFOで競合整理が行われてFIFOに順次書き込まれ、ソフトウェアによる競合処理を全く必要としない。受信側は、分散共有メモリ内の要求登録FIFOを順次読み出すのみで、実際に発生した要求のみを効率よく取り出すことができ、従来のような多数の監視点を順次サーチしていくための処理オーバーヘッドを大幅に削減することができる。このように、例えば同一メモリエリアへの同時アクセス等、従来、マルチプロセッサシステムで大きな問題となっていたプロセッサ間のリソース競合を回避することができるので、多数のプロセッサを組み合わせた超並列コンピュータシステムを構築した場合でも、プロセッサの台数に応じた大きな処理能力を持たせることが可能である。

【0123】

【発明の効果】以上説明したように、本発明によれば、カーネル等のシステムソフトウェアのオーバーヘッドを削減できるとともに、転送遅延時間が短く、処理効率の高いメッセージ転送を実現することが可能である。

【図面の簡単な説明】

【図1】本発明の一実施例を示すプロセッサモジュールの内部構成図である。

【図2】従来のマルチプロセッサシステムの構成図である。

【図3】従来のプロセッサ間メッセージ通信方法を示すシーケンスチャートである。

【図4】本発明の一実施例を示すマルチプロセッサシステムの構成図である。

【図5】本発明における分散共有メモリのデータ配置を示す図である。

【図6】同じく、分散共有メモリのデータ配置の残りを示す図である。

55

【図 7】同じく、分散共有メモリのデータ配置の残りを示す図である。

【図 8】本発明におけるモジュール ID 管理部の構成図である。

【図 9】本発明におけるパケット送信レジスタと、パケット受信レジスタのデータフォーマット図である。

【図 10】本発明におけるシステム内メッセージ通信の概要を示す図である。

【図 11】本発明におけるシステム間メッセージ通信の概要を示す図である。

【図 12】本発明におけるオブジェクト ID のフォーマット図である。

【図 13】本発明における送信ディスクリプタと送信処理要求のメモリ配置図である。

【図 14】本発明における送信処理要求のフォーマット図である。

【図 15】本発明における各種制御データの構造を示す図である。

【図 16】本発明におけるレディキューの構造を示す図である。

【図 17】本発明におけるシステムルーチングテーブルと PM ルーチングテーブルの構造図である。

【図 18】本発明におけるシステム内 PM 内通信のタイムチャートである。

【図 19】本発明におけるシステム内 PM 間通信のタイムチャートである。

【図 20】本発明におけるシステム間 PM 無中継通信のタイムチャートである。

【図 21】本発明におけるシステム間 PM 中継通信のタイムチャートである。

【図 22】本発明におけるアプリケーションオブジェクトの起動、終了処理のフローチャートである。

【図 23】本発明における MB 捕捉処理のフローチャートである。

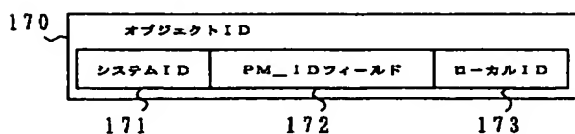
【図 24】本発明における MB 解放処理のフローチャートである。

【図 25】本発明における送信処理のフローチャートである。

【図 26】本発明における受信処理のフローチャートである。

【図 27】本発明におけるエンキュー処理のフローチャートである。

【図 12】



56

*ートである。

【図 28】本発明におけるデキュー処理のフローチャートである。

【図 29】本発明における I/O 配信処理のフローチャートである。

【図 30】本発明における INC 送信処理のフローチャートおよびアダプタ送信ディスクリプタの構造図である。

【図 31】本発明における INC 受信処理のフローチャートである。

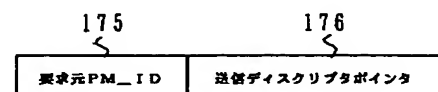
【図 32】同じく、INC 受信処理の残りのフローチャートである。

【図 33】本発明におけるアダプタ受信ディスクリプタの構造図である。

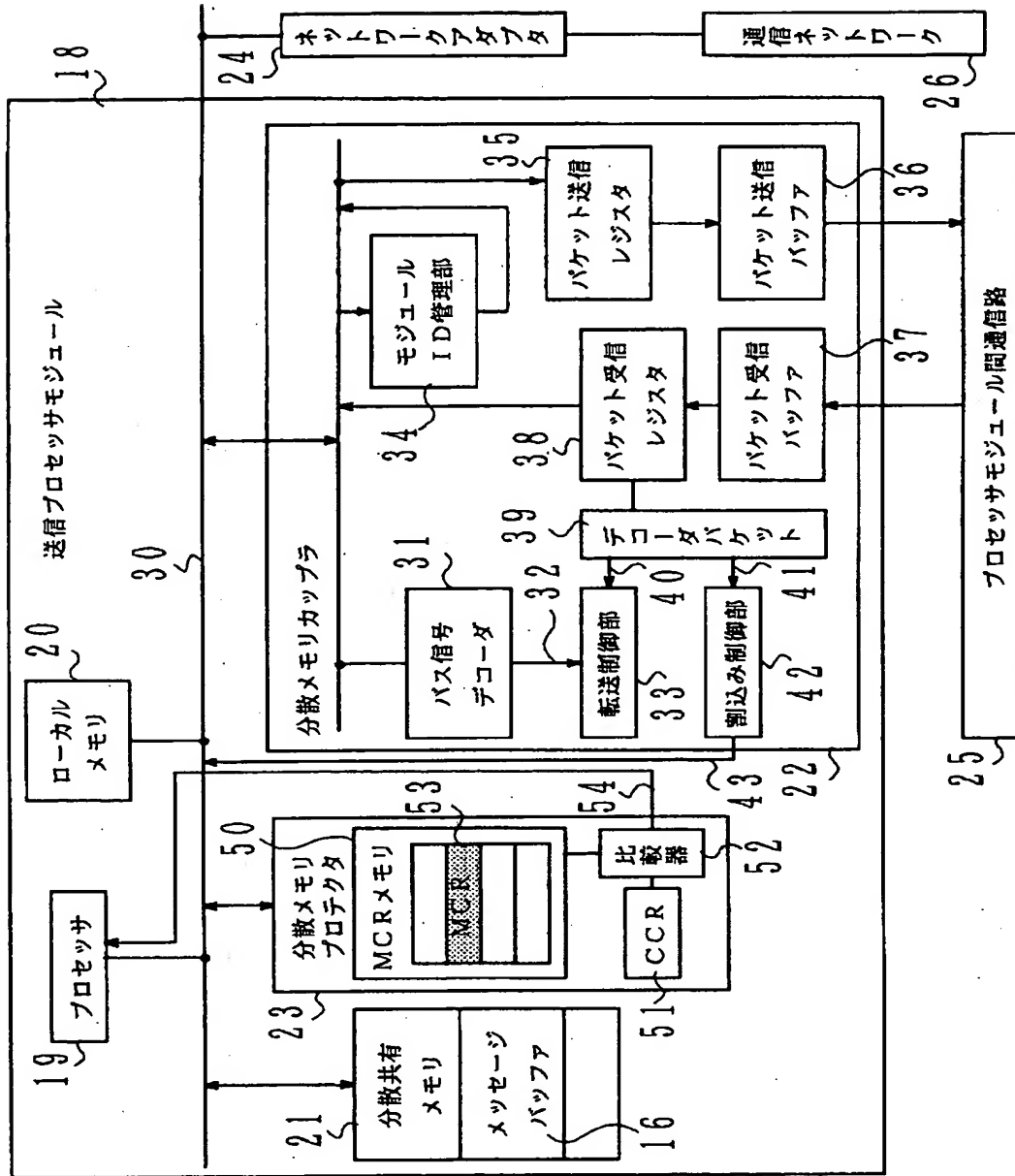
【符号の説明】

17・・・マルチプロセッサシステム、
 18-1, 18-2, 18-3・・・プロセッサモジュール、
 19-1, 19-2, 19-3・・・プロセッサ、
 20-1, 20-2, 20-3・・・ローカルメモリ、
 21-1, 21-2, 21-3・・・分散共有メモリ、
 22-1, 22-2, 22-3・・・分散メモリカップラ、
 23-1, 23-2, 23-3・・・分散メモリプロテクタ、
 24-1, 24-2, 24-3・・・ネットワークアダプタ、
 25・・・プロセッサモジュール間通信路、
 26・・・通信ネットワーク、34・・・モジュール ID 管理部、
 35・・・パケット送信レジスタ、36・・・パケット送信バッファ、
 37・・・パケット受信バッファ、38・・・パケット受信レジスタ、
 50・・・MCR メモリ (メモリキーバリティレジスタメモリ)、
 51・・・CCR (カレントキーバリティレジスタ)、
 53・・・MCR、60・・・モジュール ID ディレクトリ、
 61・・・CAM (Content Addressable Memory)、
 62・・・データメモリ部、

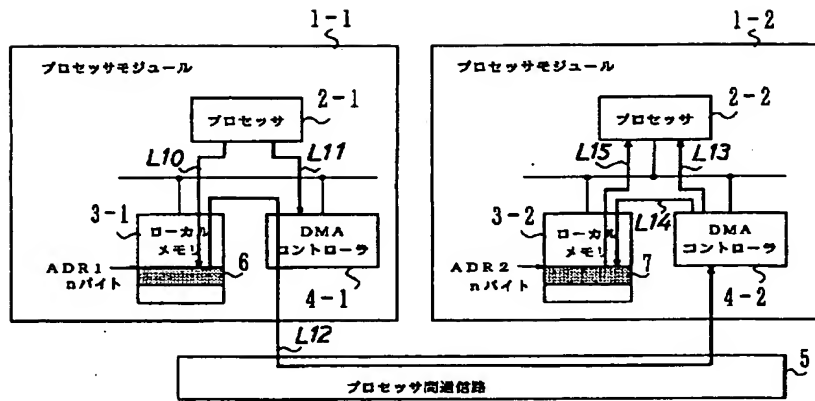
【図 14】



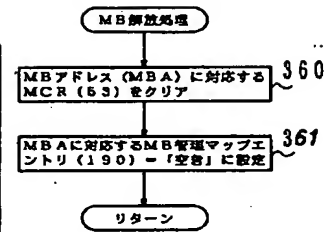
【図1】



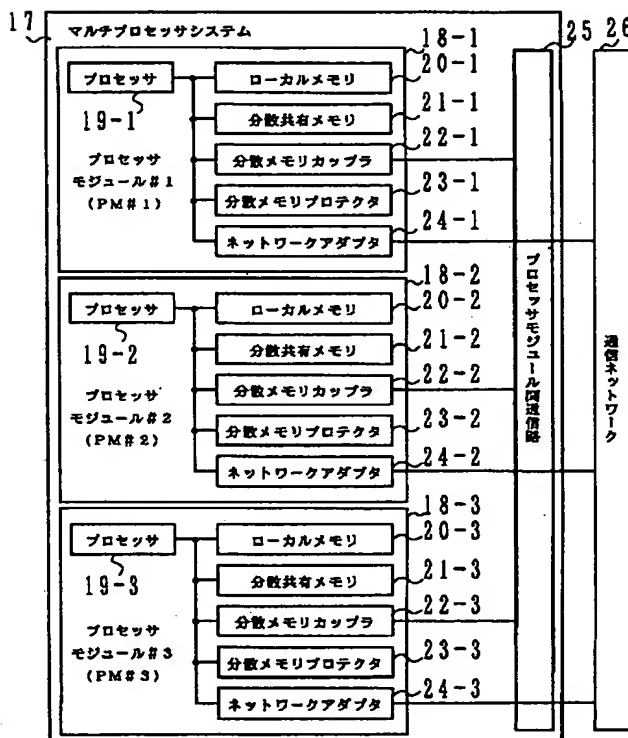
【図2】



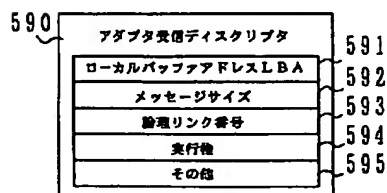
【図24】



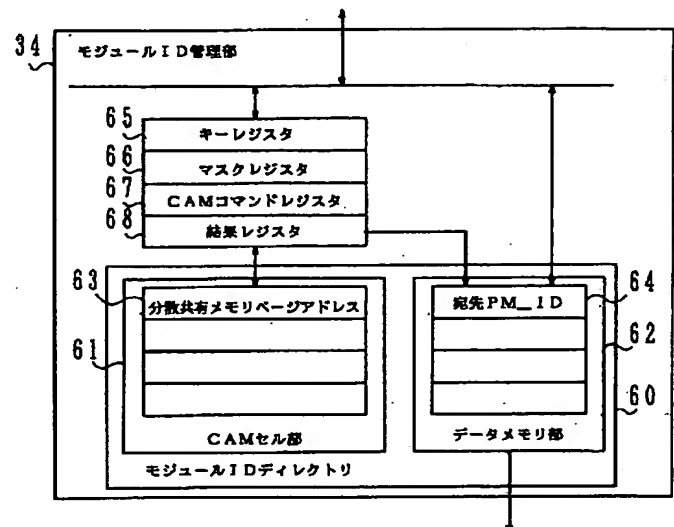
【図4】



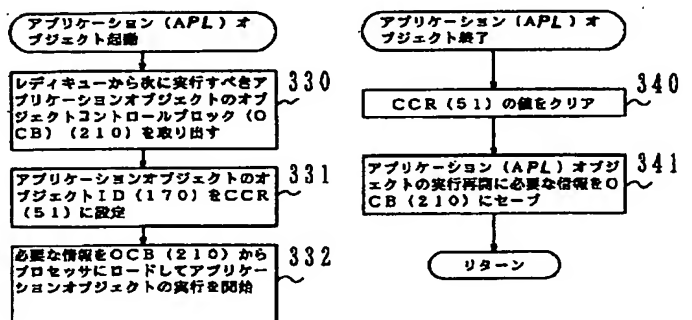
【図33】



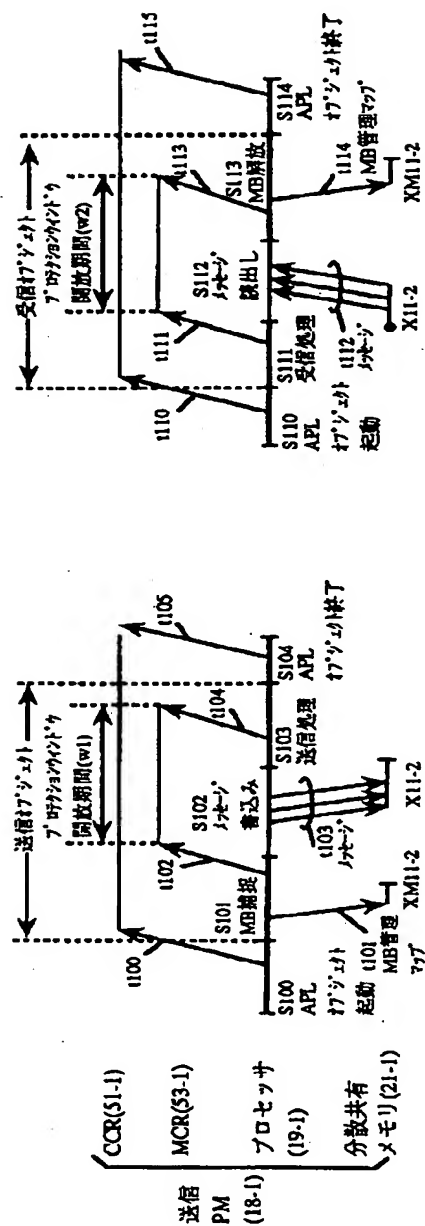
【図8】



【図22】



【图 18】



【図 5】

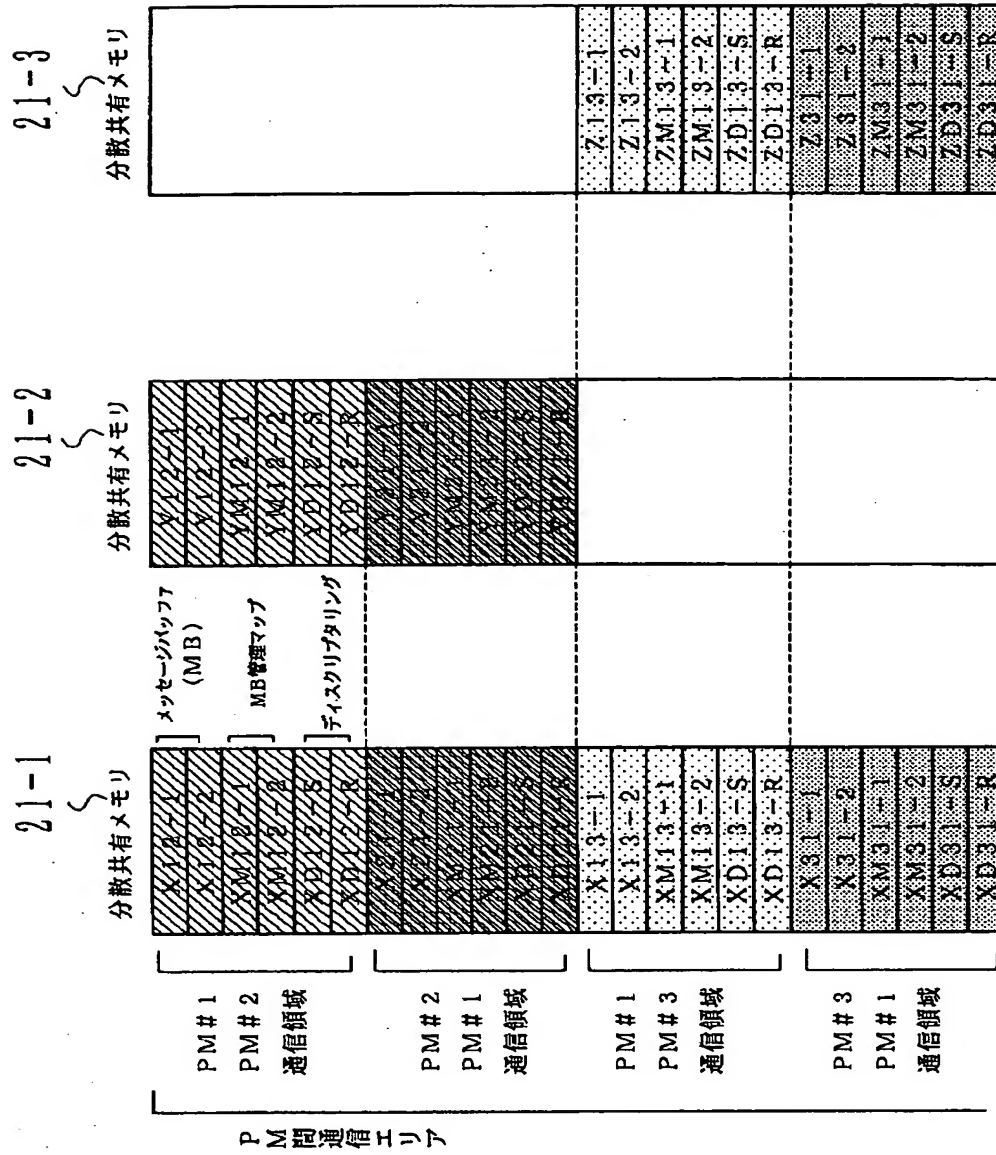
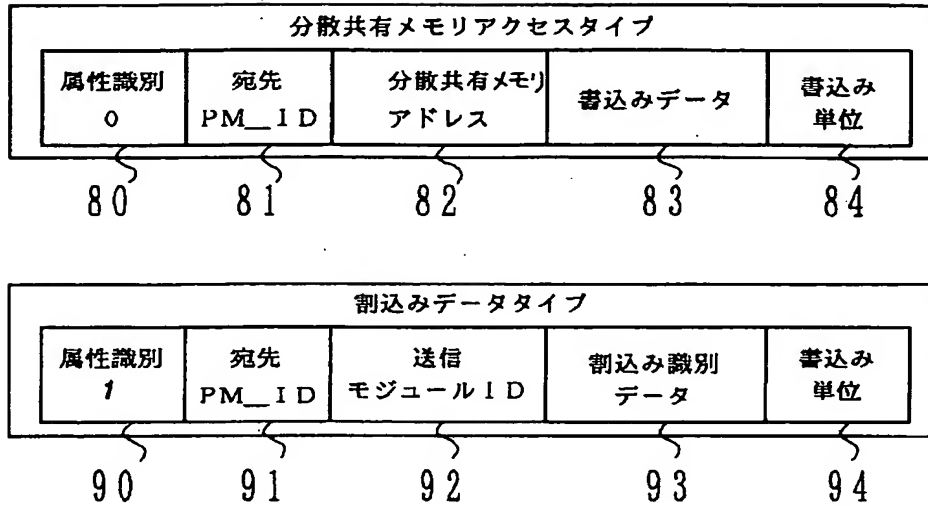


Figure 1 is a schematic diagram of a memory system. It is organized into three main horizontal sections, each representing a different memory type or access method:

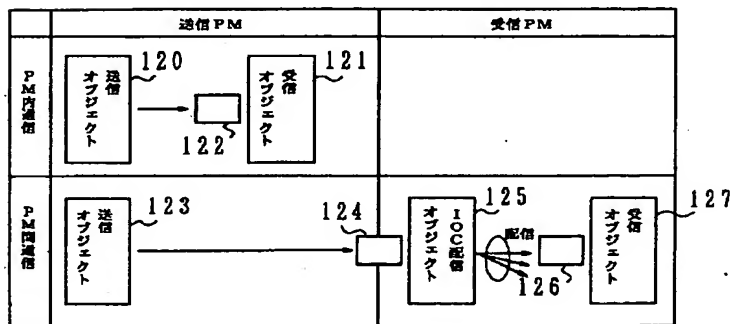
- PM#1 (Top Section):** Labeled "PM#1内通信領域" (Communication Area within PM#1) on the left. It contains a "分散共有メモリ" (Distributed Shared Memory) area with blocks labeled X111-1, X111-2, XM111-1, XM111-2, XD111-S, and XD111-R.
- PM#2 (Middle Section):** Labeled "PM#2内通信領域" (Communication Area within PM#2) on the left. It contains a "分散共有メモリ" (Distributed Shared Memory) area with blocks labeled Y222-1, Y222-2, YM222-1, YM222-2, YD222-S, and YD222-R.
- PM#3 (Bottom Section):** Labeled "PM#3内通信領域" (Communication Area within PM#3) on the left. It contains a "分散共有メモリ" (Distributed Shared Memory) area with blocks labeled Z333-1, Z333-2, ZM333-1, ZM333-2, ZD333-S, and ZD333-R.

At the bottom right, there is a "FIFO通信エリア" (FIFO Communication Area) with blocks labeled ZF1-S, ZF1-R, ZF2-S, ZF2-R, ZF3-S, and ZF3-R. These blocks are connected to the PM#3 section via a "FIFO" (First In, First Out) communication path.

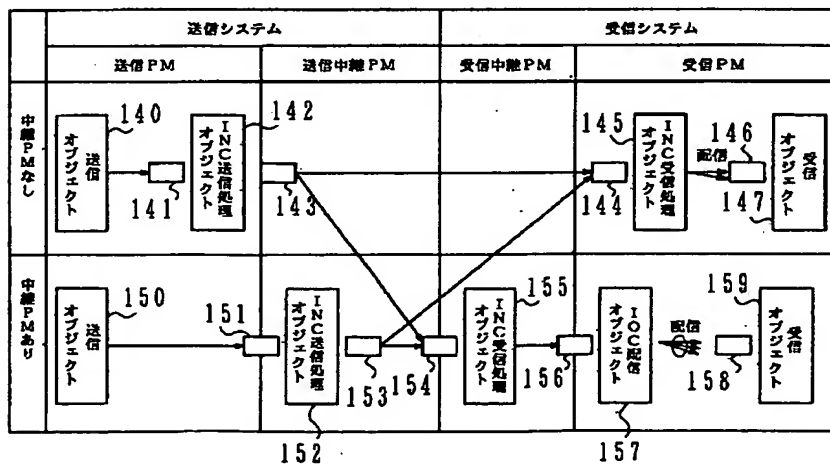
【図 9】



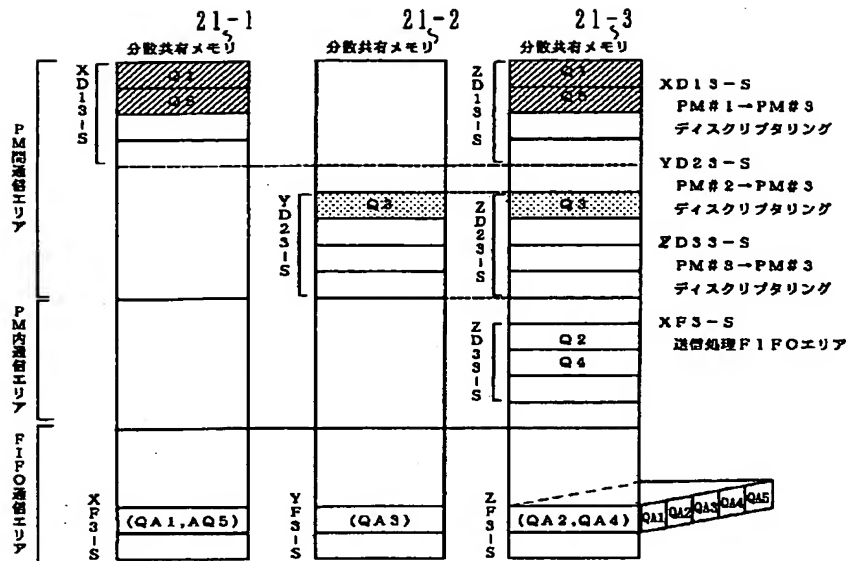
【図 10】



【図 11】



【図13】



【図15】

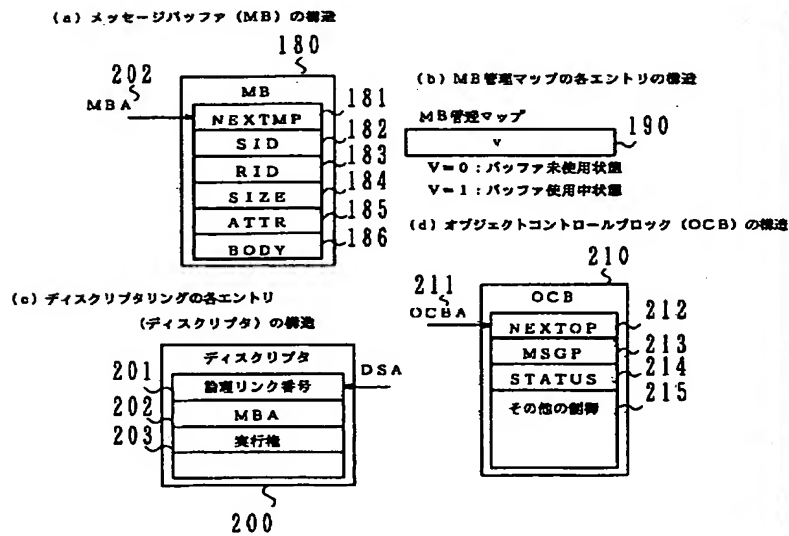
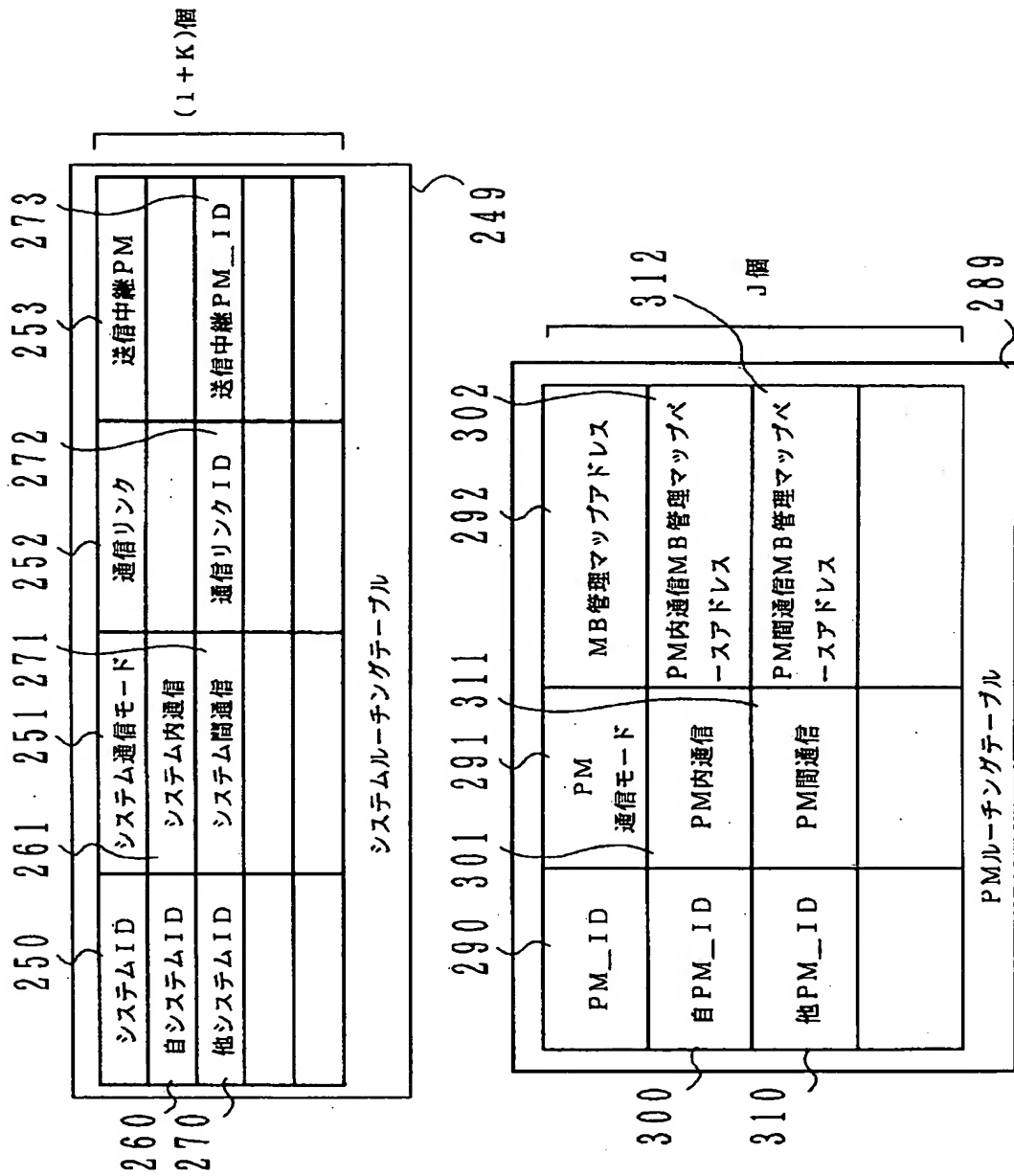


Figure 1 is a block diagram of a multi-processor system. A central block labeled "READYP" is connected to a vertical bus labeled "メッセージバユー" (Message Bus). The bus connects to two columns of blocks. The left column contains blocks labeled "OCB_1", "MB_11", and "MB_12". The right column contains blocks labeled "OCB_J", "MB_J1", "MB_J2", and "MB_J3". Each block contains sub-labels: "OCB_1" and "OCB_J" have "NEXTMP" and "MSGP"; "MB_11", "MB_J1", "MB_J2", and "MB_J3" have "NEXTMP". Arrows indicate data flow from the bus to the blocks and from the blocks back to the bus.

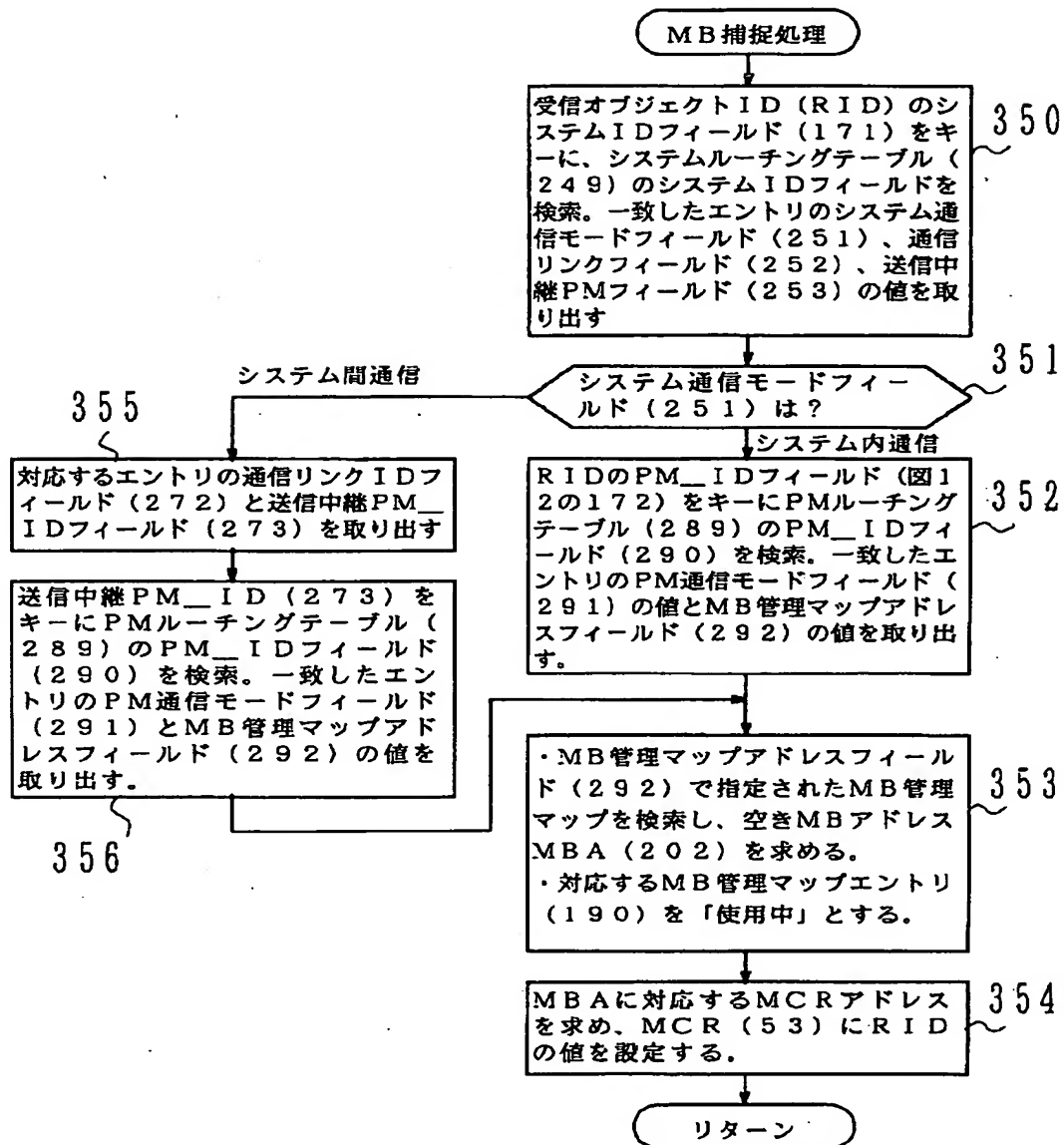
[illegible]

【図 17】

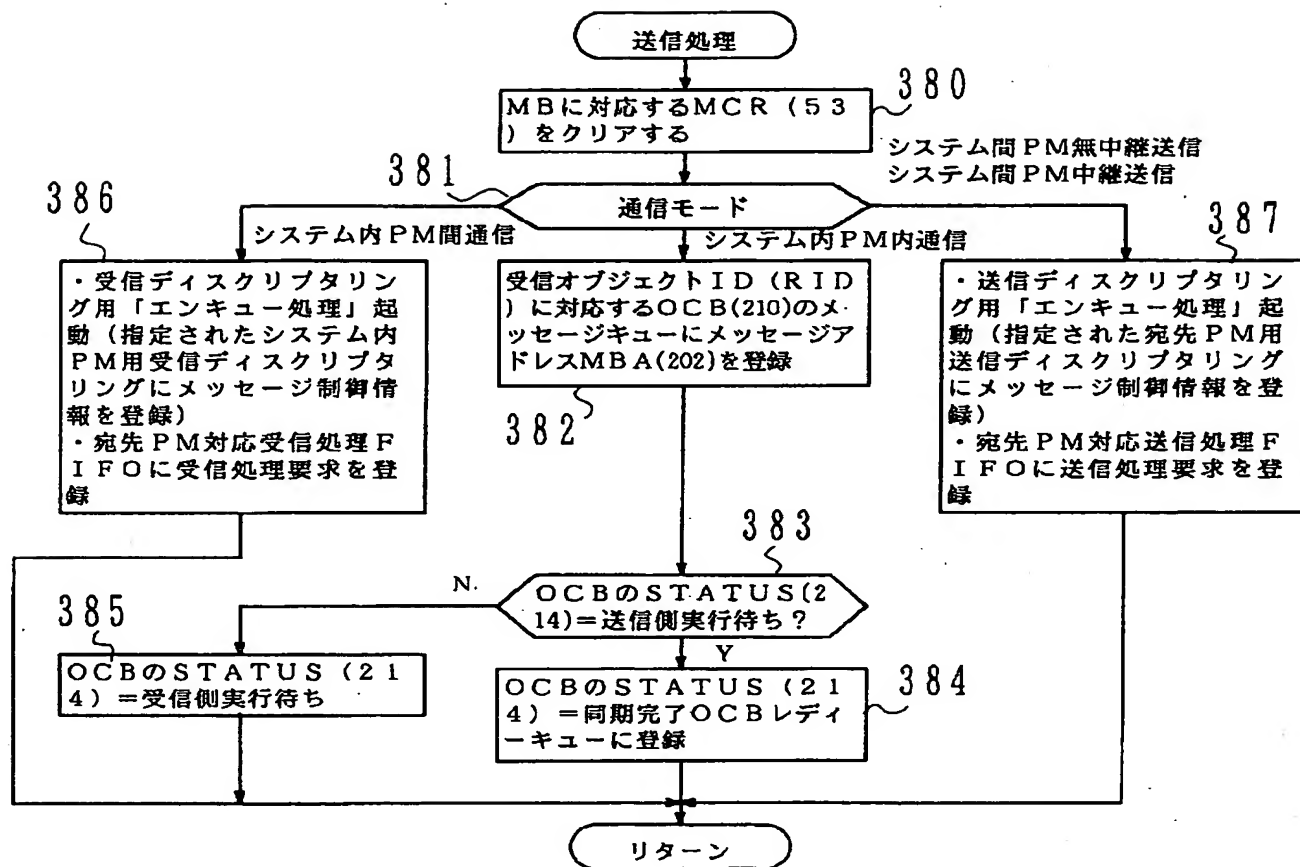


[illegible]

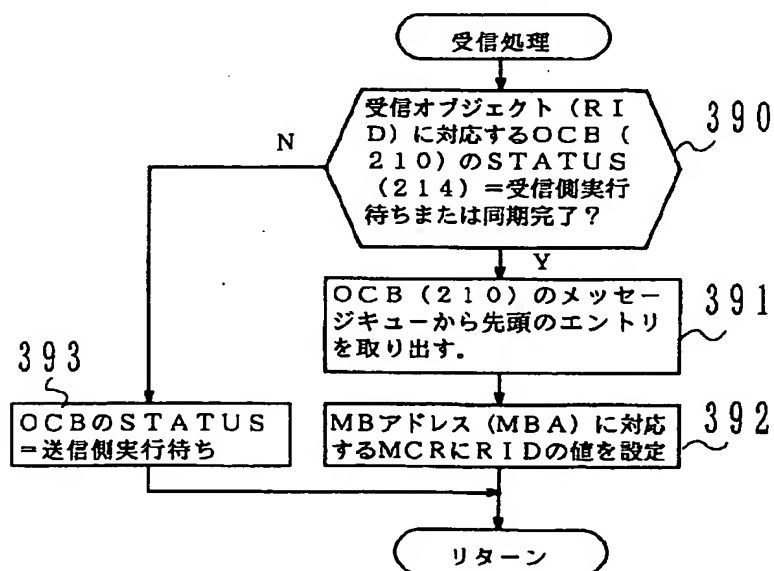
【図 23】



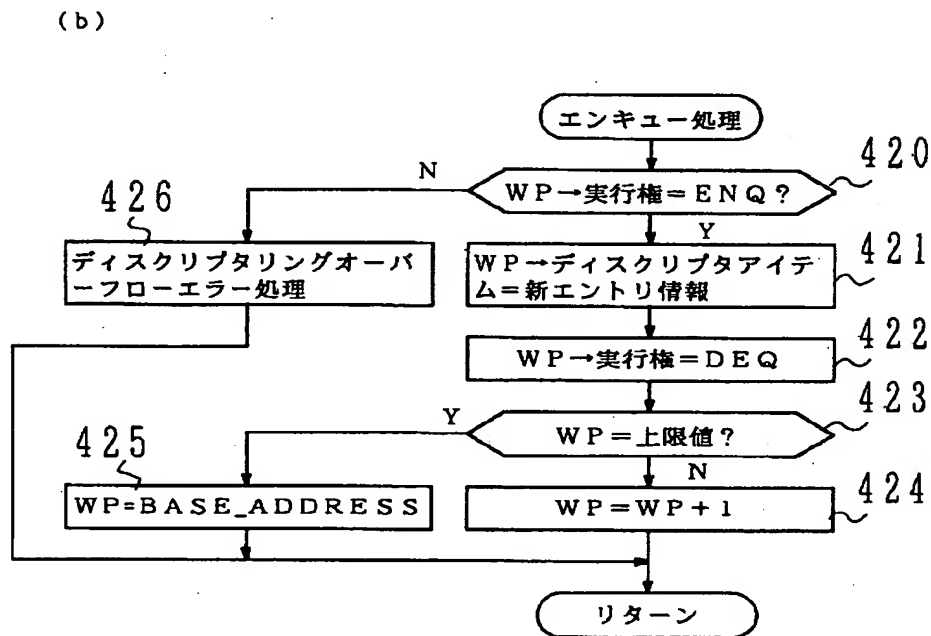
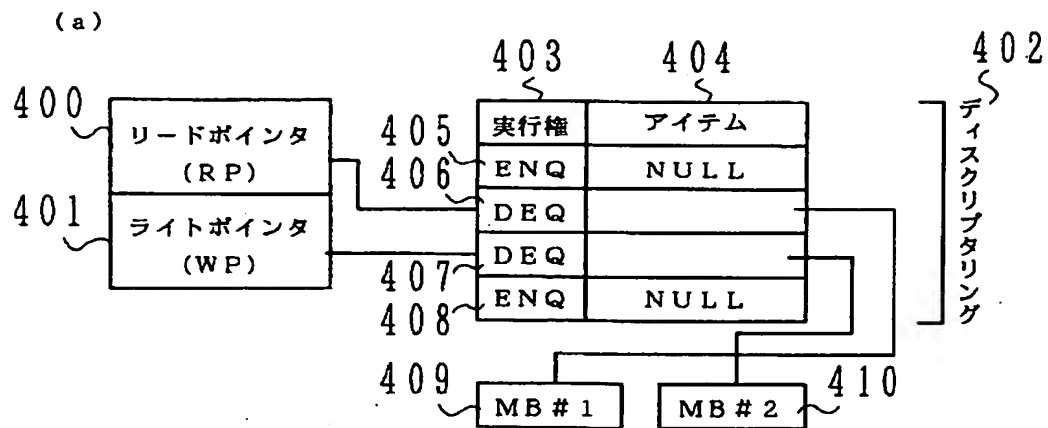
【図 25】



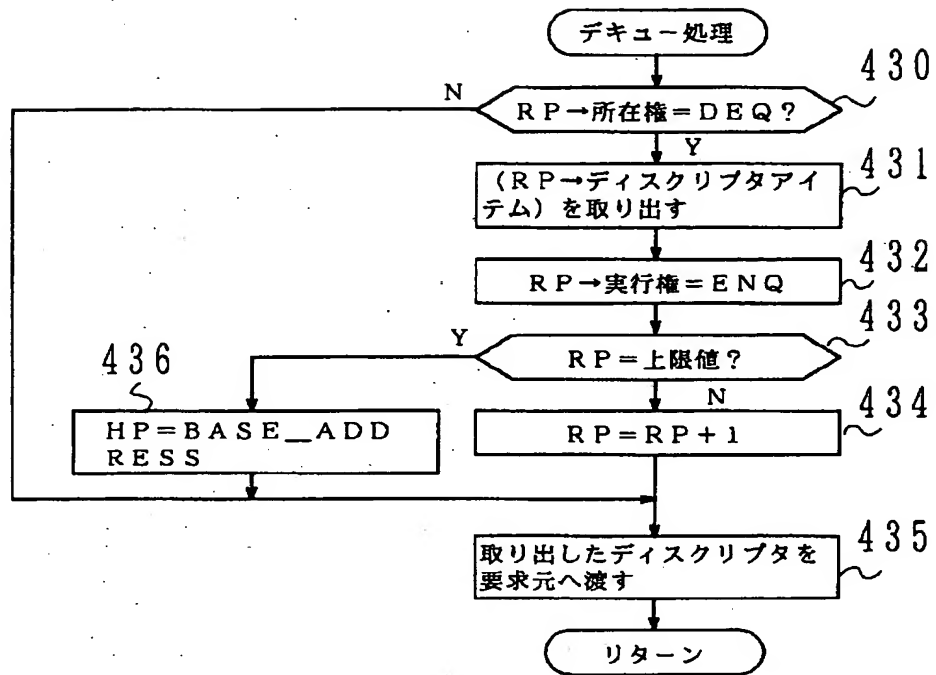
【図 26】



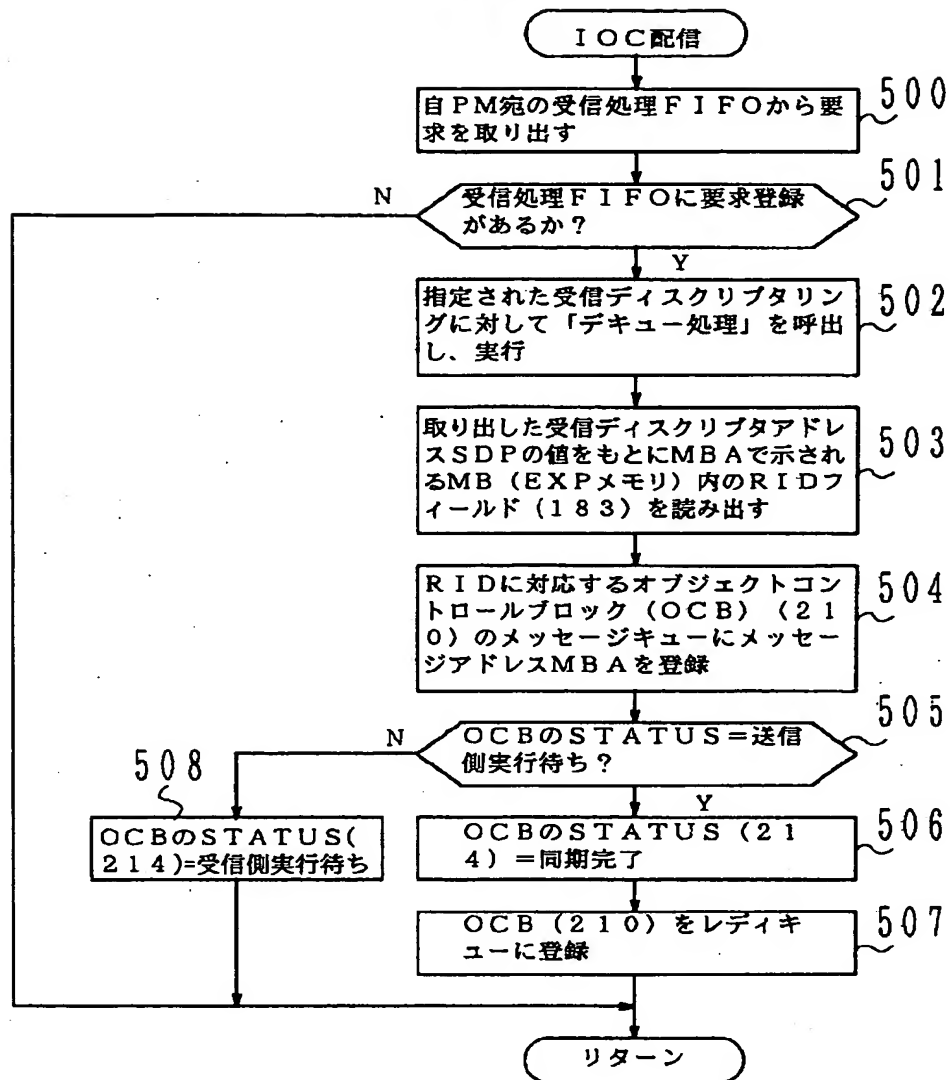
【図 27】



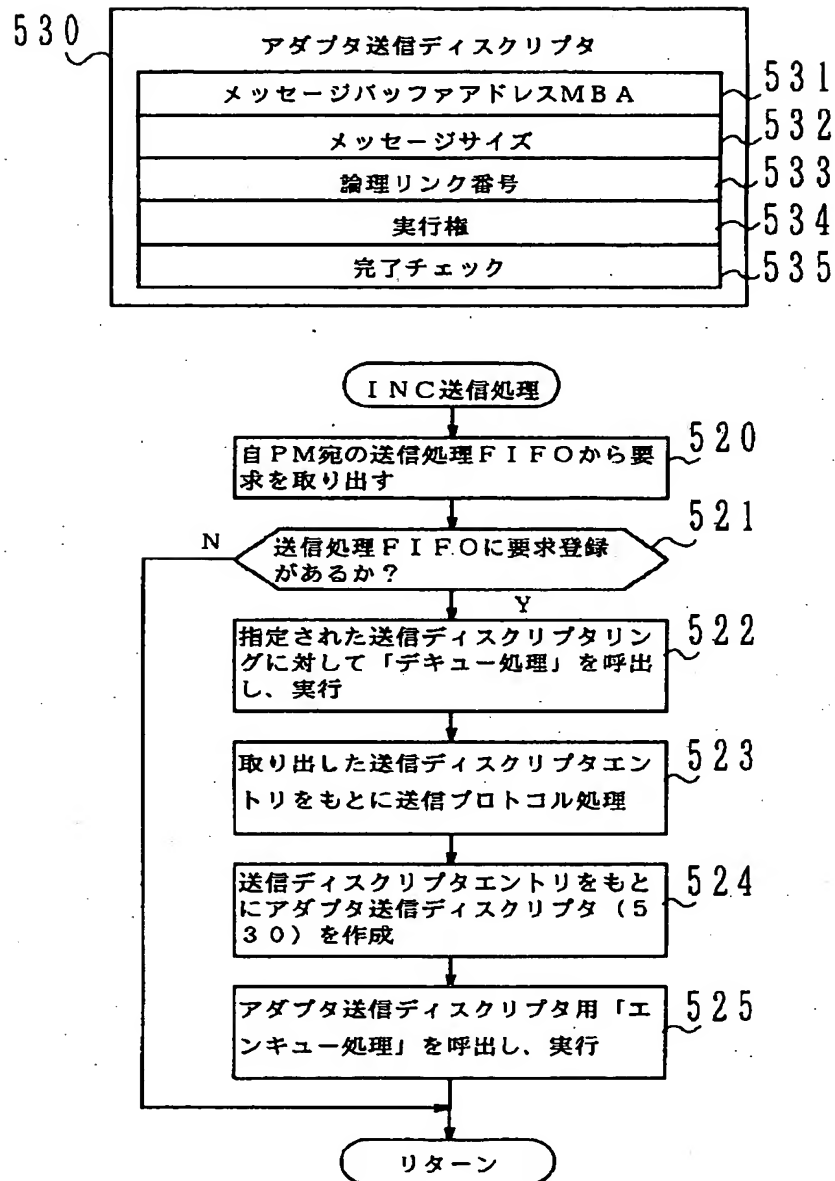
【図 28】



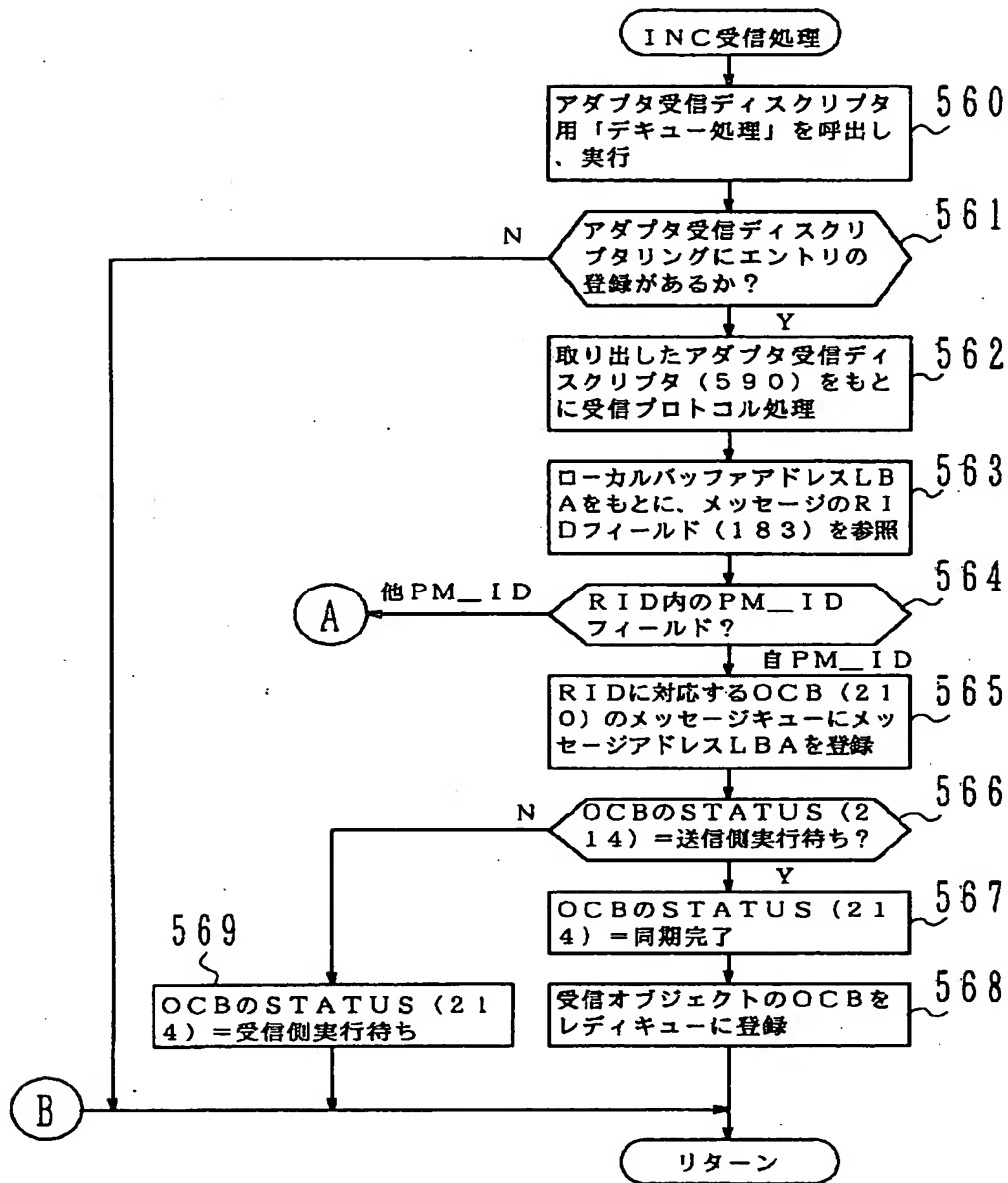
【図29】



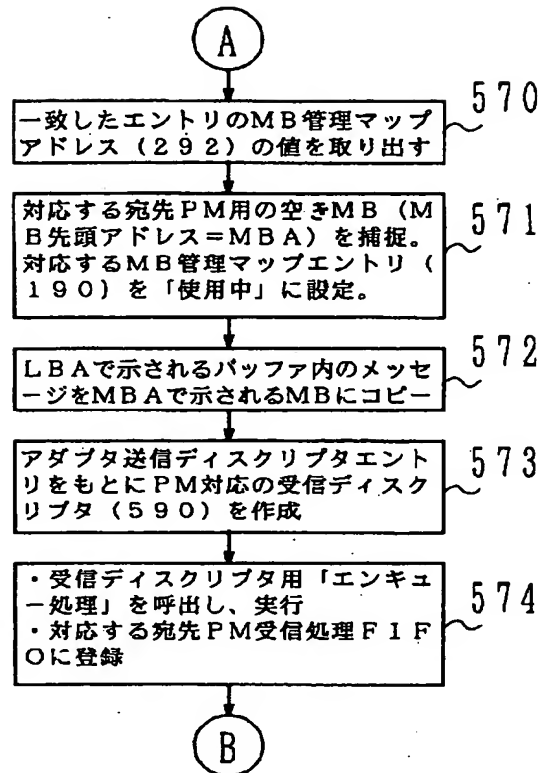
【図 30】



【図 31】



【図 3 2】



【手続補正書】

【提出日】平成7年2月6日

【手続補正1】

【補正対象書類名】明細書

【補正対象項目名】0015

【補正方法】変更

【補正内容】

【0015】図5～図7のエリア内の記述に示すように、分散共有メモリ21上の各エリアを識別するために、 M_{ijk} のような識別名を用いている。ここで、 M はプロセッサモジュール対応のMB/MB管理マップ/MBディスクリプタリング/FIFO通信エリアの識別記号であり、 $PM(18-1)$ に対応するMBは X 、そのMB管理マップは XM 、MBディスクリプタリングは XD 、FIFO通信エリアは XF と表現される。同様に、 $PM(18-2)$ に対応するMBは Y 、そのMB管理マップは YM 、MBディスクリプタリングは YD 、FIFO通信エリアは YF 、 $PM(18-3)$ に対応するMBは Z 、そのMB管理マップは ZM 、MBディスクリプタリングは ZD 、FIFO通信エリアは ZF のように名称が付けられている。また、 i, j は、それぞ

れ送信PMの識別番号($PM\ ID$)と受信PMの識別番号($PM\ ID$)を表わしており、 $PM\#1(18-1)$ 、 $PM\#2(18-2)$ 、 $PM\#3(18-3)$ に対応してそれぞれ1, 2, 3の番号が付与されている。ただし、例外として、FIFO通信エリア XF 、 YF 、 ZF は値 j のみを有し、値 i を持っていない。その理由としては、FIFO通信エリアが値 j で表わされる受信側プロセッサモジュール対応に分割されており、値 i で表わされる各送信プロセッサモジュール対応に分割されており、値 i で表わされる各送信プロセッサモジュール間で共通に使用するため、エリアの指定情報として値 i を必要としないためである。

【手続補正2】

【補正対象書類名】明細書

【補正対象項目名】0018

【補正方法】変更

【補正内容】

【0018】ディスクリプタリングエリアも、MBと同様に、宛先 $PM-ID$ が同一の複数のMBディスクリプタを分散共有メモリの連続するアドレスロケーショ

ンに割り付けており、割り付けた全体をディスクリプタリングと呼ぶ。『リング』と呼ばれている理由は、ディスクリプタリングに含まれる複数のディスクリプタを若いアドレス順にサイクリックに使用していくからである。送信側の各ディスクリプタリングと同一アドレスのエリアが、受信プロセッサモジュールの分散共有メモリにも配置される。ディスクリプタリングは、送信オブジェクトからのメッセージ制御情報を他のPMあるいは他のシステムに引き継ぐための『送信ディスクリプタリング』と、他のPMあるいは他のシステムから受信したメッセージ制御情報を受信オブジェクトに引き継ぐための『受信ディスクリプタリング』に分けられる。ディスクリプタリングもMBやMB管理マップの場合と同じように、送信側と受信側の同一アドレス間でペアを構成する。ディスクリプタリング内のディスクリプタとMBとの対応は、実行時にカーネルにより動的に決定される。具体的には、例えばPM# 1 (18-1) からPM# 2 (18-2) にメッセージを送信する場合に、PM (18-1) のカーネルが空きのMBプールの中から適当なMB (例えばX12-2) を捕捉し、次に対応する送信ディスクリプタリング (例えばX12-S) を選択し、その中の『未使用』の最若アドレスのディスクリプタにMBアドレスを登録する。このような方法でディスクリプタとMBとの対応が動的に決定される。なお、送信ディスクリプタリングXD12-Sとペアを組む送信ディスクリプタリングはYD12-Sである。

【手続補正3】

【補正対象書類名】明細書

【補正対象項目名】0019

【補正方法】変更

【補正内容】

【0019】図7におけるFIFO通信エリアは、送信オブジェクトからのメッセージ受信処理要求を他のPMまたは他のシステムに通知するための『送信処理FIFOエリア』と、他のPMまたは他のシステムから受信したメッセージ受信処理要求を受信オブジェクトに通知するための『受信処理FIFOエリア』に分けられる。このうち、メッセージ処理要求送信側の分散共有メモリは、通常のRAM (ランダムアクセスメモリ) で構成されるが、メッセージ処理要求受信側の分散共有メモリは、FIFOで構成されている。例えば、PM# 1 (18-1) 上のPM# 1 (18-1) 宛受信処理FIFOエリアXF1-Rは、他のPMからのメッセージ受信処理要求を記憶するためにFIFOメモリで構成されているが、PM# 2上のPM# 1 (18-1) 宛受信処理FIFOエリアYF1-Rおよび、PM# 3上のPM# 1 (18-1) 宛受信処理FIFOエリアZF1-Rは、いずれもRAMで構成されている。この理由としては、受信処理要求を書き込む側のPM (PM# 2とPM# 3) が受信処理要求をそれぞれ受信処理FIFOエリアYF1-RとZF1-Rに同時に書き込んだ場合、それらが同時に受信処理要求を読み出す側のPM (PM# 1) の受信処理FIFOエリアXF1-Rに到着するので、これらの受信処理要求を全て蓄積するためにFIFOメモリを使用するからである。

フロントページの続き

(72)発明者 田中 聡
東京都千代田区内幸町1丁目1番6号 日
本電信電話株式会社内

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☐ FADED TEXT OR DRAWING
- ☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☒ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☒ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.